

T_EX Live 指南—2026

Karl Berry 编写

<https://tug.org/texlive/>

2026 年 2 月

目录

1	简介	4
1.1	T _E X Live	4
1.2	操作系统支持	4
1.3	T _E X Live 的基本安装	5
1.4	关于安全的考虑	5
1.5	获得帮助	5
2	T_EX Live 概览	6
2.1	T _E X Collection: T _E X Live, MacT _E X, MiK _T _E X, CTAN	6
2.2	T _E X Live 的顶层目录	6
2.3	预定义的 texmf 目录树概览	7
2.4	T _E X 的扩展版本	8
2.5	T _E X Live 中其他值得一提的程序	9
3	安装	9
3.1	启动安装程序	9
3.1.1	Unix	10
3.1.2	macOS	10
3.1.3	Windows	11
3.1.4	Cygwin	11
3.1.5	文本化安装程序	11
3.1.6	图形化安装程序	12

3.2	执行安装程序	12
3.2.1	二进制系统菜单 (只对 Unix 适用)	13
3.2.2	选择要安装的组件	13
3.2.3	目录	14
3.2.4	选项	16
3.3	install-tl 命令行选项	17
3.3.1	-repository 参数	18
3.4	安装后的操作	18
3.4.1	Unix 下的环境变量	18
3.4.2	环境变量的全局配置	19
3.4.3	DVD 安装后的网络更新	19
3.4.4	XeTeX 和 LuaTeX 的系统字体配置	19
3.4.5	ConTeXt LMTX 和 MKIV	20
3.4.6	集成本地与个人宏文件	20
3.4.7	集成第三方字体	21
3.5	测试安装是否成功	21
3.6	卸载 TeX Live	23
3.7	其他可下载软件的链接	23
4	特殊安装	24
4.1	用户共享 (或跨机器) 安装	24
4.2	便携 (USB) 安装	24
5	tlmgr: 管理你的安装	25
5.1	tlmgr 的 GUI 界面	25
5.2	tlmgr 命令行使用示例	25
6	有关 Windows 平台的说明	28
6.1	针对 Windows 的特征	28
6.2	Windows 上附加的软件	29
6.3	使用外部的 Perl、Tcl/Tk 和 Ghostscript 程序	29
6.4	User Profile 目录相当于主目录	30
6.5	Windows 注册表	30
6.6	Windows 权限	30

目 录	3
6.7 增加 Windows 和 Cygwin 下的最大内存量	31
7 Web2C 用户指南	31
7.1 Kpathsea 路径搜索	32
7.1.1 路径的来源	33
7.1.2 配置文件	33
7.1.3 路径展开	34
7.1.4 默认展开	34
7.1.5 大括号展开	34
7.1.6 子目录展开	35
7.1.7 texmf.cnf 文件中的特殊字符简要说明	35
7.2 文件名数据库	35
7.2.1 文件名数据库	35
7.2.2 kpsewhich: 独立的路径搜索	36
7.2.3 使用举例	36
7.2.4 调试操作	38
7.3 运行时选项	40
7.4 \$TEXMFDOTDIR	41
8 致谢	41
9 发行历史	43
9.1 过去	43
9.1.1 2003	44
9.1.2 2004	44
9.1.3 2005	46
9.1.4 2006–2007	46
9.1.5 2008	47
9.1.6 2009	48
9.1.7 2010	48
9.1.8 2011	49
9.1.9 2012	49
9.1.10 2013	50
9.1.11 2014	50

1 简介	4
9.1.12 2015	51
9.1.13 2016	52
9.1.14 2017	52
9.1.15 2018	53
9.1.16 2019	54
9.1.17 2020	55
9.1.18 2021	56
9.1.19 2022	57
9.1.20 2023	59
9.1.21 2024	60
9.2 当前: 2025	62
9.3 未来	64
10 翻译说明	64

1 简介

1.1 T_EX Live

本文档描述 T_EX Live 软件的主要功能和特性, T_EX Live 是 T_EX 及其相关程序在 GNU/Linux 及其他类 Unix 系统、macOS 和 Windows 系统下的一套发行版。你可以通过下载、从 DVD 或其他途径获得 T_EX Live。

T_EX Live 包含了 T_EX, L^AT_EX 2_ε, ConT_EXt, METAFONT, MetaPost, B_IB_T_EX 和许多其他程序的可执行文件; 种类繁多的宏包、字体和文档; 并支持世界上许多不同种语言的排版。

文档末尾的第 9 节 (第 43 页) 将简要介绍这一版 T_EX Live 的重要改变。

1.2 操作系统支持

T_EX Live 为多种基于 Unix 的平台提供了二进制文件, 包括 GNU/Linux、macOS、和 Cygwin。它所包含的源代码还可在没有二进制文件的平台上编译安装。

至于 Windows, T_EX Live 仅支持 Windows 7 及后续版本。Windows Vista 或许还可以继续工作, 但 T_EX Live 不能在 Windows XP 或者更早的版本上运行。T_EX Live 包含了 64 位的 Windows 可执行文件。

除了 T_EX Live 以外, Windows 和 macOS 用户还有其它的选择, 请参考第 2.1 节。

1.3 T_EX Live 的基本安装

你可以用 DVD 或通过网络来安装 T_EX Live (<https://tug.org/texlive/acquire.html>)。网络安装程序本身很小, 而且它会从网上下载一切所需的东西。

DVD 的安装程序可以把 T_EX Live 安装到你的本地磁盘上, 但你不能直接从 T_EX Collection DVD (或者 T_EX Collection 和 T_EX Live 的 .iso 镜像) 上运行 T_EX Live。你可以在 USB 盘上安装一套可运行的版本 (参见 4.2)。安装方法将在下面的章节介绍 (p.9), 这里提供一个快速入门:

- 对于 Unix, 安装脚本是 `install-tl`; 在 Windows 则需要调用 `install-tl-windows`。安装程序将在指定了 `-gui` 选项的情况下以图形界面模式工作 (这是 Windows 下的默认模式), 或在指定了 `-gui=text` 选项的情况下以文本模式工作 (这是其它系统下的默认模式)。
- 安装完成后可以得到一个名为 `tlmgr` 的程序: ‘T_EX Live Manager’。和安装程序一样, 它可以在 GUI 模式或文本模式下运行。你不但可以用它来安装或卸载软件包, 还可以用来完成各种配置工作。

1.4 关于安全的考虑

尽我们所知, 核心的 T_EX 程序本身是 (而且一直是) 非常健壮的。然而, 尽管每个人都尽力而为, 大家贡献给 T_EX Live 的程序却不一定能达到同样的程度, 所以, 当运行不受信任的输入程序时, 你应该一直保持警惕; 为了提高安全性, 请使用单独的子目录或者 chroot。

在 Windows 系统上特别需要注意对当前目录的文件进行检查, 因为通常情况下, Windows 会忽视搜索路径而优先使用当前目录下的程序。这给许多可能的攻击带来了方便。我们已经堵住了很多漏洞, 但毫无疑问还有漏洞存在, 尤其是在第三方程序方面。因此, 我们建议检查当前目录下的可疑文件, 尤其是可执行文件 (二进制文件或者脚本)。一般情况下, 它们不应该存在, 而且更不应该在处理文档时生成。

最后, T_EX (及其相关程序) 在处理文档时是可以写文件的, 这个功能可能被各方滥用, 所以我们再一次强调在新的子目录下处理未知文档是最保险的选择。

另一个关乎安全的方面是确保下载的内容与它被创建时一致。如果安装了 `gpg` (GNU Privacy Guard) 程序, `tlmgr` 程序 (第 5 节) 会自动对现在内容进行加密验证。`gpg` 不和 T_EX Live 一起分发, 但可以参见 <https://texlive.info/tlpgp/> 了解更多信息。

1.5 获得帮助

T_EX 社群是活跃而友好的, 几乎所有认真的提问都能得到回答。尽管如此, 这种由志愿者和业余用户组成的技术支持仍然显得不太正式, 所以, 在提问前最好做好功课。(如果你更喜欢有保障的商业性技术支持, 可以放弃 T_EX Live, 改为购买商业 T_EX 系统, 在 <https://tug.org/interest.html#vendors> 上有一份销售商的列表。)

大概按照推荐使用的顺序, 我们列出了这样一份资源列表:

起步 如果你刚刚接触 T_EX, <https://tug.org/begin.html> 页面提供了这个系统的简短介绍。

CTAN 如果你在寻找某个特定的宏包、字体、程序等等, CTAN 就是你首先该找的地方。这里是一个所有 T_EX 相关内容的巨大集合。目录表条目还可以告诉你一个宏包能否用于 T_EX Live 或 MiK_TE_X。参见 <https://ctan.org>。

T_EX FAQ T_EX FAQ 汇集了各类问题——从最基础到最晦涩的。它可以在 <https://texfaq.org> 页面找到。

T_EX 网上资源 <https://tug.org/interest.html> 页面上有许多和 T_EX 相关的链接，包括讨论这个系统方方面面的许多书籍、手册和文章。

支持网站 一般性的网络检索没有坏处。最重要的 T_EX 技术支持网站包括 L^AT_EX 社区网站 <http://latex.org/>，问答网站 <https://tex.stackexchange.com>，和邮件列表 <https://lists.tug.org/texhax>。它们的内容归档中有多年以来的提问和回答供你搜索。

提问 如果你还是找不到答案，可以在 <https://latex.org> 和 <https://tex.stackexchange.com/> 网站上提问，或者发送邮件到 texhax@tug.org（提问并不需要订阅）。不过，在提问前，请先阅读 FAQ 上的这一条：<https://texfaq.org/FAQ-askquestion>，它能提高你获得有用回答的可能性。

T_EX Live 技术支持 如果你需要报告错误，或者提出对 T_EX Live 的发行、安装或文档的建议和意见，可以使用 tex-live@tug.org 这个邮件列表。不过，如果问题是针对 T_EX Live 中包含的某个特定程序，那最好还是写信给这个程序的维护者或邮件列表。用 `--help` 参数来运行程序一般都能得到用来报告其 bug 的地址。

另一方面，你也不妨帮助其他有问题的朋友。上述所有资源对所有人开放，所以请随时参与，开始阅读，在你能力所及的范围内提供帮助。

2 T_EX Live 概览

这个小节描述的是 T_EX Live 以及它所属的 T_EX Collection 的内容。

2.1 T_EX Collection: T_EX Live, MacT_EX, MiK_T_EX, CTAN

T_EX Collection DVD 包含了以下内容：

T_EX Live 是一个完全跨平台的 T_EX 系统，它可以安装在本地磁盘上。主页：<https://tug.org/texlive/>。

MacT_EX 针对 macOS，在 T_EX Live 的基础上增加了原生的 macOS 的安装程序和一些其它的 Mac 应用程序。主页：<https://tug.org/mactex/>。

MiK_T_EX 另一个用于 Windows、GNU/Linux 和 macOS 的完全跨平台的 T_EX 发行版（但 DVD 上只包含 Windows 的二进制文件）。它有一个集成的包管理器，可以根据需要从互联网安装缺失的组件。主页：<https://miktex.org/>。

CTAN 一份 CTAN 仓库的快照（<https://ctan.org/>）。CTAN 并不一定遵循 T_EX Live 的版权协议，因此在分发或修改时要格外小心。

2.2 T_EX Live 的顶层目录

这是一个简短的 T_EX Live 发行版顶层目录的列表及描述。

bin T_EX 系统程序，按平台组织。

readme-*.dir 关于 T_EX Live 的多种语言的简介和有用的链接，同时有 HTML 和纯文本版本。

source 所有程序的源代码，包括主要的基于 Web2C 的 T_EX 发行版。

texmf-dist 最主要的文件树，见下文的 TEXMFDIST。

tlpkg 用来维护安装程序所用到的脚本、程序和数据，以及对 Windows 的特殊支持。

至于文档，顶层目录下的 `doc.html` 文件中提供的完整链接会有帮助。几乎所有内容的文档（宏包、格式文件、字体、程序手册、man page、Info 文件等）都在 `texmf-dist/doc` 目录下，不管它被放在哪里，你都可以使用 `texdoc` 程序来寻找这些文档。

T_EX Live 本身的文档在 `texmf-dist/doc/texlive` 目录下，有以下这些语言的版本：

- 捷克/斯洛伐克语: `texmf-dist/doc/texlive/texlive-cz`
- 德语: `texmf-dist/doc/texlive/texlive-de`
- 英语: `texmf-dist/doc/texlive/texlive-en`
- 法语: `texmf-dist/doc/texlive/texlive-fr`
- 意大利语: `texmf-dist/doc/texlive/texlive-it`
- 日语: `texmf-dist/doc/texlive/texlive-ja`
- 波兰语: `texmf-dist/doc/texlive/texlive-pl`
- 俄语: `texmf-dist/doc/texlive/texlive-ru`
- 塞尔维亚语: `texmf-dist/doc/texlive/texlive-sr`
- 西班牙语: `texmf-dist/doc/texlive/texlive-es`
- 简体中文: `texmf-dist/doc/texlive/texlive-zh-cn`

2.3 预定义的 `texmf` 目录树概览

本小节列出了系统中用于指定 `texmf` 目录的所有预定义变量及其用途，以及 *T_EX Live* 的默认布局。`tlmgr conf` 命令可以列出这些变量的值，这样你可以很容易找到它们和你所安装到的目录名称的对应关系。

所有这些目录树，包括个人的，都应该遵循 *T_EX* 目录结构 (*T_EX* Directory Structure, TDS, <https://tug.org/tds>)，包括其中所有细小的子目录，否则文件就可能找不到。第 3.4.6 节 (第 20 页) 有更详细的介绍。这里描述的顺序和搜索的优先级相反，也就是说，列在更靠后的目录树会被更优先搜索。

TEXMFDIST 这个目录树包含几乎所有原有发行版本的文件——配置文件、脚本、宏包、字体等等。（主要的例外是每个平台的可执行文件，它们存储在同级的 `bin/` 目录下。）

TEXMFSYSVAR 给 `texconfig-sys`、`updmap-sys`、`fmtutil-sys` 和 `tlmgr` 使用的（全站的）目录树，用于存储（缓存）运行时的数据，如格式文件和生成的映射文件。

TEXMFSYSCONFIG 给 `texconfig-sys`、`updmap-sys` 和 `fmtutil-sys` 使用的（全站的）目录树，用于存储修改过的配置数据。

TEXMFLOCAL 系统管理员用来安装供整个系统使用的额外的或更新过的宏包、字体等的目录树。

TEXMFHOME 用户用来安装供他们自己独立使用的额外的或更新过的宏包、字体等的目录树。这个变量根据不同的用户选择不同的个人目录。

TEXMFVAR 给 `texconfig`、`updmap-user` 和 `fmtutil-user` 使用的（个人的）目录树，用于存储（缓存）运行时的数据，如格式文件和生成的映射文件。

TEXMFCONFIG 给 `texconfig`、`updmap-sys` 和 `fmtutil-sys` 使用的（个人的）目录树，用于存储修改过的配置数据。

TEXMFCACHE 给 `ConTEXt MkIV` 和 `LuaLATEX` 使用的目录树，用于存储（缓存）运行时的数据。缺省为 **TEXMFSYSVAR**；若该目录不可写，则使用 **TEXMFVAR**。

默认的目录结构：

全系统根目录 可以包含多个 *T_EX Live* 版本：（在 Unix 下默认是 `/usr/local/texlive`）
2024 上一个版本。

2025 当前版本。

```
bin
    i386-linux GNU/Linux 二进制文件 (32 位)
    ...
    universal-darwin macOS 二进制文件
    x86_64-linux GNU/Linux 二进制文件 (64 位)
    windows Windows 二进制文件 (64 位)
texmf-dist    TEXMFDIST 和 TEXMFMAIN
texmf-var     TEXMFSYSVAR, TEXMFCACHE
texmf-config  TEXMFSYSCONFIG
texmf-local  TEXMFLOCAL 用来存放在不同版本间共享的数据。
用户主 (home) 目录 ($HOME 或 %USERPROFILE%)
.texlive2024 给上个版本的, 个人生成和配置的数据。
.texlive2025 给这个版本的, 个人生成和配置的数据。
texmf-var     TEXMFVAR, TEXMFCACHE
texmf-config  TEXMFCONFIG
texmf TEXMFHOME 个人的宏包文件, 等等。
```

2.4 T_EX 的扩展版本

Knuth 所开发的原始 T_EX 本身已经冻结了, 仅修改少量的错误。它在 T_EX Live 中仍然作为 tex 程序出现, 在可预见的未来也仍然如此。T_EX Live 包括了一些建立在 T_EX 基础上的扩展程序 (也称为 T_EX 引擎):

ε-T_EX 为 T_EX 增加了一套新的原语 (primitive)。(包括宏展开, 字符扫描, mark 的分类, 额外的调试功能, 等等) 以及用于双向排版的 T_EX--X_ET 扩展模式。在默认情况下, ε-T_EX 100% 兼容原始的 T_EX。参见 `texmf-dist/doc/etex/base/etex_man.pdf`。

pdfT_EX 在 ε-T_EX 扩展的基础上构建, 在 DVI 输出之外增加对 PDF 输出的支持, 以及许多非输出相关的扩展。它是许多常见格式, 如 etex, latex 或 pdflatex, 调用的程序。它的主页在 <http://www.pdfptex.org/>, 文档见 `texmf-dist/doc/pdftex/manual/pdftex-a.pdf`。可以在 `texmf-dist/doc/pdftex/samplepdftex/samplepdf.tex` 找到展示部分功能的例子。

LuaT_EX 增加了对 Unicode 输入以及 OpenType/TrueType 字体和系统字体的支持。它集成的 Lua 语言解释器 (<https://lua.org>) 为许多棘手的 T_EX 问题提供了解决方案。当以 `texlua` 命令执行时, 它就像一个独立的 Lua 解释器一样运行。其网站是 <http://www.luatex.org>, 参考手册在 `texmf-dist/doc/luatex/luatex.pdf`。

(e)(u)pT_EX 原生支持日语排版需求; pT_EX 是基本引擎, 而 epT_EX 添加了 ε-T_EX 功能, 而 upT_EX 添加了 Unicode 支持。

XeT_EX 通过标准的第三方库, 增加了对 Unicode 输入以及 OpenType/TrueType 字体和系统字体的支持。参见 <https://tug.org/xetex>。

Ω (Omega) 基于 Unicode (16 位字符集), 因而同时支持处理世界上几乎所有的语言文字。它同时还支持所谓的 ‘Ω 翻译过程’ (Ω Translation Processes, OTPs), 用于对任意输入进行复杂的变换操作。

Omega 现在已经不作为独立的程序出现在 T_EX Live 中了; 改为只支持 Aleph:

Aleph 结合了 Ω 与 ε-T_EX 扩展。参见 `texmf-dist/doc/aleph/base`。

2.5 T_EX Live 中其他值得一提的程序

这里给出 T_EX Live 中其他的一些常用程序:

bibtex, biber 参考文献支持。

makeindex, upmendex, xindex, xindy 索引支持。

dvips 将 DVI 转换为 PostScript。

dvipdfmx 将 DVI 转换为 PDF, 是 (前面提到过的) pdfT_EX 的一套替换方案。

xdvi X Window 系统下的 DVI 阅读器。

dviconcat, dviselect 从 DVI 文件中剪切和粘贴页面。

psselect, psnup, ... PostScript 实用程序。

pdfjam, pdfjoin, ... PDF 实用程序。

context, mtxrun ConT_EXt 和 PDF 处理工具。

htlatex, ... tex4ht: (L^A)T_EX 到 HTML (还有 XML 以及 DocX 等其他格式) 的转换器。

3 安装

3.1 启动安装程序

首先请找来一张 T_EX Collection 的 DVD, 或者下载 T_EX Live 的网络安装程序。参见 <https://tug.org/texlive/acquire.html> 以了解更多关于获得这个软件的信息和方法。

网络安装程序, .zip 或 .tar.gz 格式: 从 CTAN 的 `systems/texlive/tlnet` 下载压缩包; 链接 <http://mirror.ctan.org/systems/texlive/tlnet> 应该能重定位到一个附近的、保持更新的镜像。你可以下载同时支持 Unix 和 Windows 的 `install-tl.zip` 或者小得多、但只支持 Unix 的 `install-unx.tar.gz`。解压后, `install-tl` 和 `install-tl-windows.bat` 就会出现在 `install-tl` 子目录中。

网络安装程序, Windows .exe: 和上述一样从 CTAN 下载, 然后双击。它会作为第一阶段的安装程序和解压工具启动; 见图 1。它提供了两个选项: “Install” 和 “Unpack only”。

T_EX Collection DVD: 打开 DVD 的 `texlive` 子目录。在 Windows 下, 安装程序通常在插入 DVD 后就自动启动了。要获得 DVD 的话, 可以加入一个 T_EX 用户组织 (非常推荐这么做, <https://tug.org/usergroups.html>) 或是单独购买 (<https://tug.org/store>)。又或者是自己从 CTAN 下载 ISO 镜像刻录 DVD。在从 DVD 或者 ISO 安装后, 如果还希望从 Internet 获得持续的更新, 请看第 3.4.3 节。

无论安装程序的来源如何, 所运行的都是相同的安装程序。两者最显著的区别在于, 使用网络安装程序时, 最终得到的是当前可用的软件包。这与 DVD 和 ISO 镜像不同, 后者在主要公开发布版本之间不会更新。

如果你需要通过代理服务器下载, 使用 `~/.wgetrc` 文件或设置环境变量来配置 Wget 的代理设置 (https://www.gnu.org/software/wget/manual/html_node/Proxies.html), 或者等效地设置你所使用的下载工具。如果你是从 DVD 或者 ISO 镜像来安装就没有关系了。

下一小节将更详细地介绍安装程序的启动。

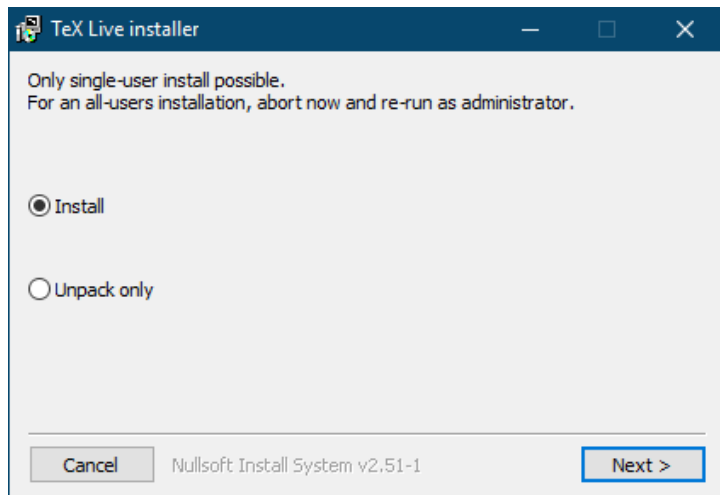


图 1: Windows .exe 安装程序的第一阶段. 点击 Install 选项将出现如图 3 所示的窗口.

3.1.1 Unix

下面的内容中, > 表示命令提示符; 用户输入显示成 **bold** 的样子。install-tl 是一个 Perl 脚本。在 Unix 兼容的系统下最简单的启动它的方法是这样的:

```
> perl /path/to/installer/install-tl
```

(或者你可以直接运行 `/path/to/installer/install-tl` 一旦它有可执行属性, 或者先 `cd` 到这个目录中, 等等; 我们不会把所有这些执行方法列出来。)你可能需要扩大终端窗口才能在一屏内显示完整的文本安装程序界面 (图 2)。

要在 GUI 模式下安装 (见图 4), 你需要安装 Tcl/Tk。安装后你可以运行:

```
> perl install-tl -gui
```

原有的 `-wizard` 和 `-perlTk/-expert` 选项现在做着和 `-gui` 选项同样的事。要列出所有这些选项:

```
> perl install-tl -help
```

关于 Unix 权限: T_EX Live 安装程序将遵照你安装时的 `umask` 设置。因此, 如果你希望你的安装能被除你之外的其他用户使用, 就必须保证你设置的权限足够, 比如 `umask 002`。更多关于 `umask` 的信息请参见你自己系统的文档。

关于 Cygwin 的特殊考虑: 和其他 Unix 兼容系统不同, Cygwin 默认并没有包含一切运行 T_EX Live 安装程序所必须的程序, 见第 3.1.4 节。

3.1.2 macOS

如第 2.1 节提到的, 我们给 macOS 准备了一套独立的发行版, 叫做 MacT_EX (<https://tug.org/mactex>)。我们推荐在 macOS 使用原生的 MacT_EX 安装程序, 而不是 T_EX Live, 因为原生的安装程序做了一些针对 Mac 的调整, 尤其是方便在 macOS 下的多个 T_EX Live 发行版之间切换, 这使用了被称

为 $\text{T}_{\text{E}}\text{XDist}$ 的数据结构。

$\text{MacT}_{\text{E}}\text{X}$ 严格依赖 $\text{T}_{\text{E}}\text{X Live}$ 构建, 所以主 $\text{T}_{\text{E}}\text{X}$ 树和二进制文件也是完全一致的。不过它添加了一些用来存放 Mac 专有文档和程序的目录。

3.1.3 Windows

如果你使用的是下载了 zip 并解压后的目录, 又或者 DVD 安装程序无法自动启动了, 请双击 `install-tl-windows.bat`。

你也可以从命令行提示符下启动安装程序。下面用 `>` 表示命令提示符; 用户输入显示成 **bold** 的样子。如果你正在安装程序目录下, 只需要运行:

```
> install-tl-windows
```

或者你也可以通过绝对路径来运行, 比如:

```
> D:\texlive\install-tl-windows
```

这是对 $\text{T}_{\text{E}}\text{X Collection DVD}$ 而言的, 假定 `D:` 是光驱。图 3 展示了 GUI 安装程序的初始基本界面, 它是 Windows 下的默认形式。

要在文本模式下安装, 使用:

```
> install-tl-windows -no-gui
```

要列出所有可用的选项:

```
> install-tl-windows -help
```

如果同一目录中还包含 `install-tl-windows.exe`, 则在以上命令中添加 `.bat` 扩展名。这通常不会发生, 除非你已经在本地获取了 `tlnet` 目录的镜像。

在 $\text{T}_{\text{E}}\text{X Live}$ 安装路径中, 应避免使用非 ASCII 字符。这尤其是在 Windows 11 下可能会出现的问题, 因为它的用户被强制使用微软网络帐户。请参阅 <https://tug.org/texlive/windows.html#nonascii> 了解一些解决方法。

3.1.4 Cygwin

在开始安装之前, 如果你还没装过 `perl` 和 `wget` 软件包, 请先使用 Cygwin 的 `setup.exe` 程序安装它们。此外还推荐安装下列软件包:

- `fontconfig` [$\text{XeT}_{\text{E}}\text{X}$ 和 $\text{LuaT}_{\text{E}}\text{X}$ 需要]
- `ghostscript` [各种实用工具需要]
- `libXaw7` [`xdvi` 需要]
- `ncurses` [给安装程序提供 `clear` 命令]

3.1.5 文本化安装程序

图 2 展示了 Unix 下文本模式的主界面, 这是 Unix 下安装程序的默认界面。

```
Installing TeX Live 2025 from: ...
Platform: x86_64-linux => 'GNU/Linux on x86_64'
Distribution: inst (compressed)
Directory for temporary files: /tmp
...
Detected platform: GNU/Linux on Intel x86_64

<B> binary platforms: 1 out of 16

<S> set installation scheme: scheme-full

<C> customizing installation collections
    40 collections out of 41, disk space required: 8648 MB (free: 138718 MB)

<D> directories:
    TEXDIR (the main TeX directory):
        /usr/local/texlive/2025
    ...

<O> options:
    [ ] use letter size instead of A4 by default
    ...

<V> set up for portable installation

Actions:
<I> start installation to hard disk
<P> save installation profile to 'texlive.profile' and exit
<H> help
<Q> quit
```

图 2: 文本安装程序主界面 (GNU/Linux)

这只是一个命令行安装程序; 完全没有光标支持。例如, 你无法在复选框或输入字段之间用 Tab 上下切换。你只能在提示符下输入特定的字符 (注意大小写) 然后按下 Enter, 之后整个终端窗口就会把新的内容刷新出来。

文本化安装程序之所以这么简陋是因为它要尽可能支持最多的平台, 就算只有一个 Perl 的系统也能运行它。

3.1.6 图形化安装程序

默认的图形化安装程序的初始界面很简单, 只有几个选项; 见图 3。它可以这样启动:

```
> install-tl -gui
```

按下 Advanced 按钮可以获得文本化安装程序的大部分选项; 见图 4。

perltk/expert 和 wizard GUI 选项现在调用常规的图形化安装程序。

3.2 执行安装程序

安装程序应该不需要文档就足够明确, 但下面还是对这些选项和子菜单作一点说明。

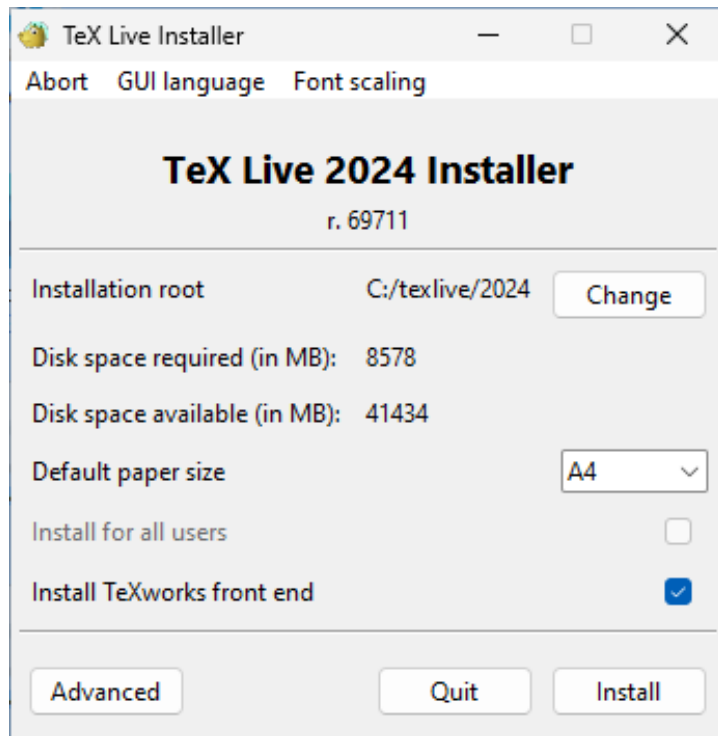


图 3: 基本安装界面 (Windows); Advanced 按钮会提供类似图 4 的界面

3.2.1 二进制系统菜单 (只对 Unix 适用)

图 5 展示了文本模式下的 `binaries` (二进制程序) 菜单。默认情况下只会安装你当前平台下的二进制程序。在这个菜单下, 你也可以选择安装其他平台的二进制程序。如果你要将 \TeX 树共享在异构网络上, 又或者将使用双启动的系统, 这会比较有用。

3.2.2 选择要安装的组件

图 6 展示了 \TeX Live 的安装方案菜单; 从这里你选择的是一套“安装方案”, 也就是对软件包集合的一个统一划分。默认的 `full` 方案会把所有可用的都装上。这是推荐方案, 不过你也可以选择其他的方案: `basic` 方案只获得 `plain` 和 \LaTeX , `small` 方案会获得比它稍微多一点的程序 (相当于 \MacTeX 中的 \BasicTeX 安装), `minimal` 方案可供测试选用, 而 `medium` 或 `teTeX` 方案则介乎其间。还有许多特殊或者专门针对特定国家的方案。

你可以使用 ‘collections’ 菜单来详细选择安装方案。(图 7 显示了 GUI 模式下修改的情况。)

Collection (安装集合) 是比 `scheme` (方案) 要更细的一层——实际上一个方案包含了多个安装集合, 而一个安装集合又包含了至少一个软件包, 然后每个软件包 (\TeX Live 中最小的组织单位) 则包含了实际的 \TeX 宏文件、字体文件等等。

如果你觉得 `collection` 菜单对安装控制还不够细, 可以在安装后使用 \TeX Live Manager (`tlmgr`) 程

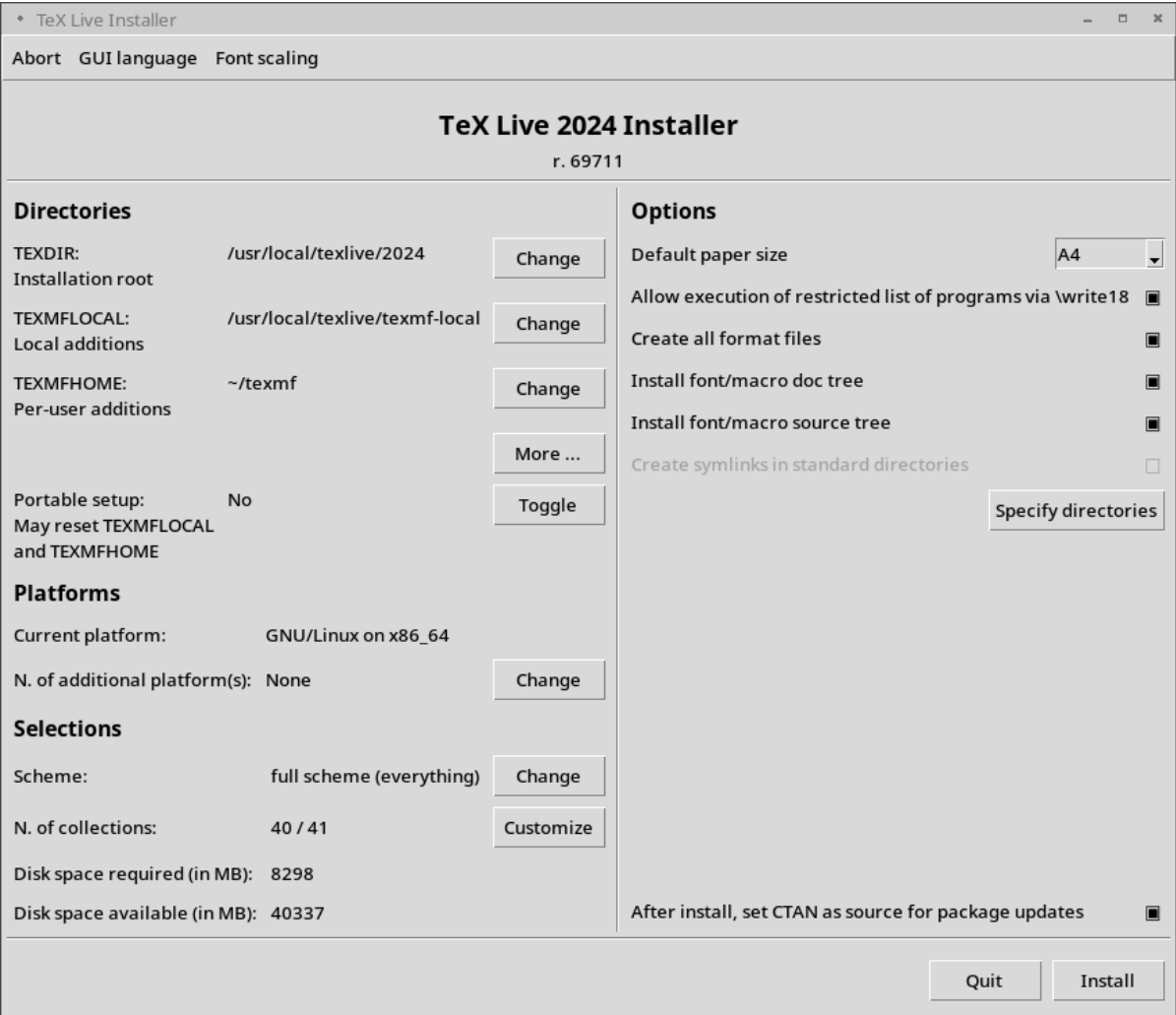


图 4: 高级 GUI 安装界面 (GNU/Linux)

序 (参见第 5 节), 它能在软件包一层控制安装。

3.2.3 目录

缺省的目录布局在第 2.3 节有过叙述, 见第 7 页。在 Unix 下默认的安装目录是 /usr/local/texlive/2025 而 Windows 下是 C:\texlive\2025。这样的安排允许你有多个并行的 T_EX Live 安装存在, 例如每个版本都有一个 (通常按年份, 如此处), 你可以通过修改搜索路径来在它们中间切换。

这个安装路径可以通过设置安装程序中的 TEXDIR 来修改。这个选项和其他选项的 GUI 界面如图 4 所示。最常见的更改它的原因要么是该分区没有足够的空间 (完整的 T_EX Live 安装需要好几 GB 的空间), 要么是没有默认位置的写权限 (虽然要安装 T_EX Live 不需要是管理员或者 root 用户, 但你至少得

```

Available platforms:
=====
a [ ] Cygwin on x86_64 (x86_64-cygwin)
b [ ] MacOSX current (10.14-) on ARM/x86_64 (universal-darwin)
c [ ] MacOSX legacy (10.6-) on x86_64 (x86_64-darwinlegacy)
d [ ] FreeBSD on x86_64 (amd64-freebsd)
e [ ] FreeBSD on Intel x86 (i386-freebsd)
f [ ] GNU/Linux on ARM64 (aarch64-linux)
g [ ] GNU/Linux on RPi(32-bit) and ARMv7 (armhf-linux)
h [ ] GNU/Linux on Intel x86 (i386-linux)
i [X] GNU/Linux on x86_64 (x86_64-linux)
j [ ] GNU/Linux on x86_64 with musl (x86_64-linuxmusl)
k [ ] NetBSD on x86_64 (amd64-netbsd)
l [ ] NetBSD on Intel x86 (i386-netbsd)
m [ ] Solaris on Intel x86 (i386-solaris)
o [ ] Solaris on x86_64 (x86_64-solaris)
p [ ] Windows (64-bit) (windows)

```

图 5: Binaries (二进制程序) 菜单

```

Select scheme:
=====
a [X] full scheme (everything)
b [ ] medium scheme (small + more packages and languages)
c [ ] small scheme (basic + xetex, metapost, a few languages)
d [ ] basic scheme (plain and latex)
e [ ] minimal scheme (plain only)
f [ ] infrastructure-only scheme (no TeX at all)
g [ ] book publishing scheme (core LaTeX and add-ons)
h [ ] ConTeXt scheme
i [ ] GUST TeX Live scheme
j [ ] teTeX scheme (more than medium, but nowhere near full)
k [ ] custom selection of collections

```

图 6: Scheme (安装方案) 菜单

对安装的目标目录有写权限)。

在 Windows 下, 通常你不需要是管理员就能创建 C:\texlive\2025 (或者, 更一般而言, 是 %SystemDrive%\texlive\2025)。

你也可以通过在运行安装程序以前修改特定的环境变量来改变安装目录 (最常见的是 `TEXLIVE_INSTALL_PREFIX` 或者 `TEXLIVE_INSTALL_TEXDIR`); 参见 `install-tl --help` 调出的文档 (<https://tug.org/texlive/doc/install-tl.html> 有在线版本) 以了解完整的列表和更多的信息。

一个合理的选择是你自己主目录下的一个子目录, 尤其在只有你一个人使用的时候。使用 ‘~’ 来表示主目录, 比如 ‘~/texlive/2025’。

我们建议在目录名称中保留年份, 这样可以让你保留多个不同版本的 TeX Live。(你可能还希望通过符号链接保留一个与版本无关的名称, 如 /usr/local/texlive-cur, 以便在测试新版本后重新指向该名称。)

在安装程序中修改 `TEXDIR` 还会同时修改 `TEXMFLOCAL`, `TEXMFSYSVAR` 和 `TEXMFSYSCONFIG`。

`TEXMFHOME` 是被推荐的用来存放个人宏文件和软件包的目录。其缺省值是 ~/texmf (Mac 下是

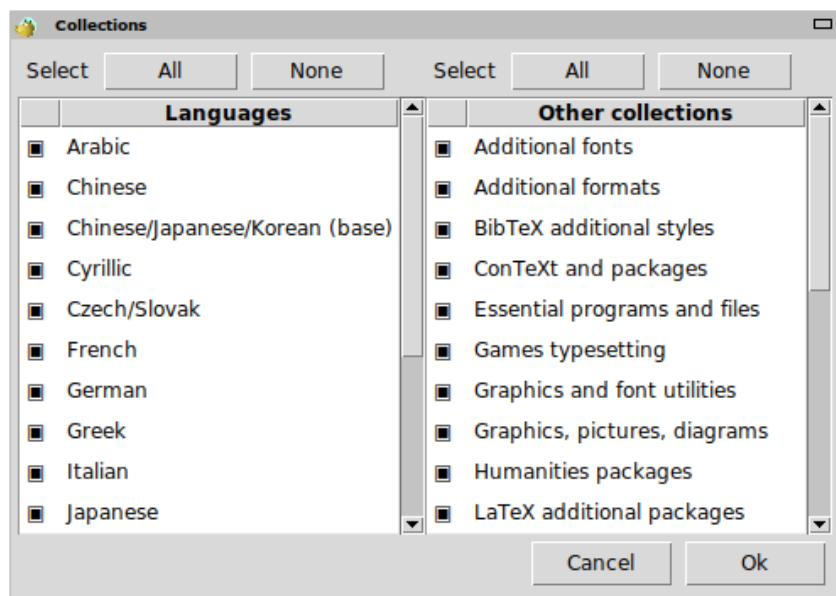


图 7: Collections (集合) 菜单

~/Library/texmf)。与 TEXDIR 不同，其中包含的 ~ 会被不加转换地写进配置文件，因为它能在 T_EX 系统运行时自动被替换为每个用户自己的主目录。在 Unix 它会被展开为 \$HOME，而 Windows 下展开为 %USERPROFILE%。多说一句：TEXMFHOME 和所有的目录树一样，都必须按照 TDS 组织，否则文件可能会找不到。

TEXMFVAR 是用来给每个用户存储大多数运行时缓存数据的。TEXMFCACHE 则为 Lua^LT_EX 和 Con-_TE_Xt MkIV 存储缓存 (见第 20 页的 3.4.5 节)；它的缺省值是 TEXMFSYSVAR，或者 (如果那不可写) TEXMFVAR。

3.2.4 选项

```
Options customization:
=====
<P> use letter size instead of A4 by default: [ ]
<E> execution of restricted list of programs: [X]
<F> create all format files: [X]
<D> install font/macro doc tree: [X]
<S> install font/macro source tree: [X]
<L> create symlinks in standard directories: [ ]
      binaries to:
      manpages to:
      info to:
<Y> after install, set CTAN as source for package updates: [X]
```

图 8: Options 菜单 (Unix)

图 8 显示了文本模式的选项菜单。关于这个菜单的每个选项:

use letter size instead of A4 by default: 缺省的纸张大小选项。当然,如果有需要,每份文档都可以并且应该单独设定一个纸张大小。

execution of restricted list of programs: 在 T_EX Live 2010 中,默认允许执行一些外部程序。这(很少的一部分)允许的程序列表在 `texmf.cnf` 中定义。参见 2010 版新闻(第 9.1.7 节)以了解更多信息。

create all format files: 我们建议保持这个选项的选定状态,以避免在动态创建格式时出现不必要的问题。更多内容请参见 `fmtutil` 文档。

install font/macro ... tree: 下载安装大部分宏包中的文档和源代码文件。建议选择。

create symlinks in standard directories: 这个选项(只对 Unix 有效的)可以省下设定环境变量的步骤。如果没有选择它,就必须把 T_EX Live 的对应目录添加到 `PATH`, `MANPATH` 和 `INFOPATH` 中。如果要创建符号链接,你需要有这些目标目录的写权限。这个选项是为了在用户已知的标准目录中创建符号链接设计的,比如 `/usr/local/bin`,这些目录并不包含任何 T_EX 文件。不要用这个选项来覆盖系统中现有的文件,比如给它指定系统目录。最保险和推荐的做法还是不要选择这个选项。

after install, set CTAN as source for package updates: 从 DVD 或 ISO 镜像安装时,这个选项被缺省启用,因为通常你会希望在此后通过 CTAN 安装一年内更新的软件包。禁用它唯一的可能是你只安装了 DVD 内容的一部分,并计划在以后扩展安装。无论如何,安装程序时和安装后的更新的软件包仓库可以根据需要分别设置,见第 3.3.1 节和第 3.4.3 节。

Windows 专有的选项,在高级的 GUI 界面中显示:

adjust searchpath 保证所有程序都可以在它们的搜索路径中找到 T_EX Live 程序目录。

add menu shortcuts 如果选择了,它会在开始菜单中创建一个 T_EX Live 子菜单。除了 ‘TeX Live menu’ 和 ‘No shortcuts’ 之外,还有 ‘Launcher entry’ 这个选项。这个选项在第 4.1 节有详细叙述。

File associations 选项包括 ‘Only new’ (创建文件关联,但不覆盖现有的关联), ‘All’ 和 ‘None’。

install T_EXworks front end

如果所有的设置已经齐备,你就可以在文本界面中按下 ‘I’,或者 GUI 界面中按下 ‘Install’ 按钮来开始安装了。安装完成后,你可以跳至第 3.4 节来了解还需要做些什么工作。

3.3 install-tl 命令行选项

输入

```
> install-tl -help
```

可以列出所有的命令行参数。你既可以用 `-` 也可以用 `--` 来指定一个参数。这里有些比较常见的:

-gui 尽可能用 GUI 模式的安装程序。它需要 Tcl/Tk 版本 8.5 或更高。在旧版 macOS 上已经分发;对于 Big Sur 及以后版本,如果你没有选择用 MacT_EX 来安装,你需要自己安装 Tcl/Tk。Tcl/Tk 已经分发于 Windows 的 T_EX Live 中。原有的 `-gui=perlTk` 和 `-gui=wizard` 仍然可以使用但它们调用的是相同的 GUI 界面。如果找不到 Tcl/Tk,安装程序就会在文本模式下出现。

-no-gui 强制使用文本模式安装程序。

-lang LL 指定安装程序界面的语言,使用标准的(通常是两个两个字符)语言代码。安装程序会尝试自己判断出合适的语言,一旦失败或判断正确的语言无法使用,程序就会使用英语替代。可以运行 `install-tl --help` 获得支持的语言列表。

- portable 为诸如 USB 盘上便携使用安装。也可以在文本安装程序中用 `v` 命令选择，或者 GUI 安装程序中选择，参见第 4.2 节。
- profile *file* 载入安装配置文件 *file* 以不需要用户干预的方式完成安装。安装程序会在安装到的 `tlpkg` 子目录中创建一个叫 `texlive.profile` 的文件。这个文件，比方说，它可以用作参数，在不同的系统下完成完全一致的安装。或者你也可以使用一个定制的配置文件的，最简单的方法是从生成的配置文件开始，修改其中的值，或者直接使用一个空文件，这将采用所有缺省的设置。
- repository *url-or-directory* 指定作为来源的软件包仓库，参见下文。
- in-place (仅为了完整性记录：除非你知道自己在做什么，否则不要使用这个选项。) 如果你已经有一份用 `rsync`, `svn`, 或者其他方式安装的 \TeX Live 副本 (参见 <https://tug.org/texlive/acquire-mirror.html>) 那么这个选项会使用现有的副本，按原样使用，并只执行必要的安装后操作。注意 `tlpkg/texlive.tlpdb` 文件可能被覆盖，你需要自己备份它。另外，包的移除必须手动完成。这个选项不能在安装程序界面中选择。

3.3.1 -repository 参数

默认的网络包仓库是由 <http://mirror.ctan.org> 自动选择的 CTAN 镜像。

如果你想要覆盖这个设置，地址可以是以 `ftp:`、`http:`、`https:`、`file:/` 开头的 URL，或者是一个普通的目录路径。(当给定一个 `http:`、`https:` 或 `ftp:` 地址时，其末尾的 `/` 字符和结尾的 `/tlpkg` 组件都会被忽略。)

比如，你可以选择这样的一个 CTAN 镜像，类似 <http://ctan.example.org/tex-archive/systems/texlive/tlnet/>，当然你应该把 `ctan.example.org/tex-archive` 替换为具体镜像的域名和特定的顶层 CTAN 路径。CTAN 的镜像列表由 <https://ctan.org/mirrors> 维护。

如果给定的参数是本地的 (无论是路径还是 `file:/` 开头的 URL)，仓库路径下的 `archive` 子目录中的压缩文件将被使用 (即使同样有未压缩的文件可用)。

3.4 安装后的操作

安装后可能需要一些额外的操作。

3.4.1 Unix 下的环境变量

如果你选择了在标准路径下创建符号连接 (在第 3.2.4 节提到)，那就不需要设置环境变量了。否则，在 Unix 系统中，你必须将自己使用的平台下的二进制程序的目录加入搜索路径中。(在 Windows 下安装程序会负责这一步。)

每个支持的平台都在 `TEXDIR/bin` 下有自己的子目录。图 5 展示了这些子目录的列表和它们对应的平台。

另外，如果还希望 `man` 和 `info` 能够被系统工具找到，你还可以把文档的 `man` 和 `Info` 目录添加至其对应的搜索路径中。在添加 `PATH` 后，`man` 页面可以被自动找到。

对于像 `bash` 这样的 Bourne 兼容的 shell 而言，以 Intel x86 下的 GNU/Linux 和 \TeX Live 默认的目录设置为例，需要修改的文件可能是 `$HOME/.profile` (或者其他由 `.profile` 载入的文件)，要添加的内容应该类似这样：

```
PATH=/usr/local/texlive/2025/bin/x86_64-linux:$PATH; export PATH
```

```
MANPATH=/usr/local/texlive/2025/texmf-dist/doc/man:$MANPATH; export MANPATH
INFOPATH=/usr/local/texlive/2025/texmf-dist/doc/info:$INFOPATH; export INFOPATH
```

对于 `csch` 或者 `tcsh`，需要修改的文件通常是 `$HOME/.cshrc`，而要添加的内容类似：

```
setenv PATH /usr/local/texlive/2025/bin/x86_64-linux:$PATH
setenv MANPATH /usr/local/texlive/2025/texmf-dist/doc/man:$MANPATH
setenv INFOPATH /usr/local/texlive/2025/texmf-dist/doc/info:$INFOPATH
```

如果你不是在 `x86_64-linux` 平台，请使用相应的平台名称；同样，如果你没有安装在默认目录中，请更改目录名称。`TEX Live` 的安装程序在安装结束时会报告要使用的完整路径。

如果你已经在启动文件中设置了 `PATH`，你可以根据需要将 `TEX Live` 目录合并到其中。

3.4.2 环境变量的全局配置

如果你想要全局地或者对于新添加到系统中的用户进行这些更改，那你就得自己动手了；因为在系统中配置这些事情的方式和位置变动太大。

我们的建议是：1) 你可能应该看看 `/etc/manpath.config` 这个文件是否存在，如果有的话，添加下面这样的内容：

```
MANPATH_MAP /usr/local/texlive/2025/bin/x86_64-linux \
    /usr/local/texlive/2025/texmf-dist/doc/man
```

然后 2) 检查 `/etc/environment` 是否定义了默认的搜索路径和其他的默认环境变量。

在每个 (Unix) 二进制目录下，我们都会创建一个名为 `man` 的符号链接关联 `texmf-dist/doc/man` 目录。有些 `man` 程序，比如 macOS 标准的 `man`，能够自动通过这个链接找到对应的手册页，这样就不必手工设置手册页路径了。

3.4.3 DVD 安装后的网络更新

如果你从 DVD 安装了 `TEX Live` 并希望从网络获取更新，需要在更新了搜索路径 (如上一节所述) 之后执行这个命令：

```
> tlmgr option repository https://mirror.ctan.org/systems/texlive/tlnet
```

这告诉 `tlmgr` 未来从就近的 CTAN 镜像获取更新。通过第 3.2.4 节介绍的选项，从 DVD 安装时会默认完成这一步。

如果自动镜像选择出现了问题，你可以从 <https://ctan.org/mirrors> 列表中自己指定一个 CTAN 镜像。使用镜像的 `tlnet` 子目录的路径，如前所示。

3.4.4 Xe_T_EX 和 Lua_T_EX 的系统字体配置

Xe_T_EX 和 Lua_T_EX 可以使用任何系统安装的字体，而不只是 `TEX` 目录树中的那些。这些系统字体 (并不属于 `TEX Live`) 常通过给出字体名来访问，例如 ‘`Liberation Serif`’，虽然也可以使用系统文件名。

相关问题是将 `TEX Live` 发行版的字体作为系统字体可用，这将使它们能通过字体名称访问。

对于 Lua_T_EX: 通过字体名访问并不需要什么特别的处理。对于 Lua_T_EX 来说, 通过同时支持 L_AT_EX 和 plain _T_EX 的 luaotfload 包, _T_EX Live 中的所有字体, 或是通过字体名, 或是通过文件名, 都同样可以访问。有新的字体时, luaotfload 包的字体名索引需要重新构建; 当尝试加载尚未知晓的字体时, 此过程会自动触发。

对于 Xe_T_EX: 在 Windows 下, 随 _T_EX Live 分发的字体会自动为 Xe_T_EX 所用 (通过运行 fc-cache 程序, 它在 Windows 作为 _T_EX Live 的一部分被提供)。在 macOS 下, 你需要查阅其他文档。对于除 macOS 之外的 Unix 系统, 步骤如下。

当 xetex 安装后 (不管是初始安装还是后来安装的), 都会在 TEXMFSYSVAR/fonts/conf/texlive-fontconfig.conf 创建一个必需的配置文件。为了让 _T_EX Live 的字体作为系统字体一样被调用,

1. 将 texlive-fontconfig.conf 文件 (通常) 复制到 /etc/fonts/conf.d/09-texlive.conf。
2. 运行 `fc-cache -fsv`。

如果你没有足够的权限执行上述操作, 或者只想把 _T_EX Live 字体提供给仅一个用户, 那么你可以这么做:

1. 将 texlive-fontconfig.conf 文件 (通常) 复制到 ~/.fonts.conf.d/09-texlive.conf, 其中 ~ 是你的主目录。
2. 运行 `fc-cache -fv`。

你可以运行 fc-list 来查看系统字体的名称。命令 `fc-list : family style file spacing` (所有这些参数都是字面字符串) 通常会显示一些很有趣的信息。

3.4.5 Con_T_EXt LMTX 和 MKIV

“旧”的 Con_T_EXt (Mark IV 或 MkIV) 和 “新的” Con_T_EXt (LMTX or MkXL) 应该在 _T_EX Live 安装后能直接运行, 而且, 只要你一直用 tlmgr 来更新, 它不应该需要任何其他处理。

Con_T_EXt 不使用 kpathsea 库或系统的 ls-R 文件; 相反, Con_T_EXt 维护了一个针对所有已知目录的每个用户的文件名数据库。运行 mktexlsr 不会直接更新这些数据库, 但是 Con_T_EXt 将在下次运行时自动重新构建它们。(_T_EX Live 特定的文件 cont-sys.mkiv 和 cont-sys.mkxl 实现了这一点。)

如果你想手动重新构建他们, 运行下面的一个或所有:

```
mtxrun --generate          # LMTX
mtxrun --luatex --generate # MkIV
```

生成的文件会被保存在 TEXMFCACHE 目录下, 它在 _T_EX Live 的默认值是 TEXMFSYSVAR;TEXMFVAR。

Con_T_EXt 会读取 TEXMFCACHE 提到的所有路径, 并写入第一个可写的路径。在读取时, 对于重复的缓存数据, 将以最后找到的匹配项为优先选择。

要了解更多信息, 参见 <https://wiki.contextgarden.net/LMTX> 和 https://wiki.contextgarden.net/Running_Mark_IV。

3.4.6 集成本地与个人宏文件

这在第 2.3 节已经顺带提到过了: TEXMFLOCAL 目录 (它的默认值是 /usr/local/texlive/texmf-local 或者 Windows 下的 %SystemDrive%\texlive\texmf-local) 就是为了存储面向整个系

统的本地字体和宏文件的; 而 `TEXMFHOME` 目录 (其默认值是 `$HOME/texmf` 或者 `%USERPROFILE%\texmf`), 则是用来存储个人的字体和宏文件的。这些目录旨在共享于各个版本之间, 并且其内容将由新的 \TeX Live 版本自动识别。因此不应该把 `TEXMFLOCAL` 的定义改得和主 \TeX Live 目录差别太大, 否则新的版本出来后你又得手动再修改。

对于这两个目录树而言, 文件都应该放到合适的 \TeX 目录结构 (TDS) 的子目录中, 参见 <https://tug.org/tds> 或者 `texmf-dist/web2c/texmf.cnf` 文件。比如一个 \LaTeX 文档类或者宏包应该放在 `TEXMFLOCAL/tex/latex` 或者 `TEXMFHOME/tex/latex` 目录下, 要不然就是它们的一个子目录下。

`TEXMFLOCAL` 目录需要一个保持更新的文件名数据库, 否则新增的文件无法被找到。你可以使用 `mktexlsr` 命令来刷新它, 也可以点击 \TeX Live Manager GUI 的 Actions 选项卡中的 ‘Update filename database’ 按钮来刷新它。

默认情况下, 这些变量的每一个都定义为所示单独的目录。这不是一个硬性规定。例如, 如果你需要在某些大型宏包的不同版本之间轻松切换, 你自己可以根据需要维护多个目录树, 这通过把 `TEXMFHOME` 设置为目录列表来实现, 记得用大括号包裹内容, 且目录之间以逗号分隔:

```
TEXMFHOME = {/my/dir1,/mydir2,/a/third/dir}
```

第 7.1.5 节进一步介绍了括号展开。

3.4.7 集成第三方字体

不幸的是, 对于 \TeX 和 \pdfTeX , 这是一个非常混乱的问题。除非你愿意深入 \TeX 安装的细节, 否则请不要涉足这个领域。 \TeX Live 已经包含了很多字体, 所以, 如果你想, 请先查看; 链接 <https://tug.org/FontCatalogue> 里面展示了主要的 \TeX 发行版中包含的几乎所有文本字体, 这些字体按不同方式分类。

如果你确实想安装自己的字体, 访问 <https://tug.org/fonts/fontinstall.html> 查看我们尽力描述的安装程序。

另外, 考虑使用 \XeTeX 或 \LuaTeX (见第 2.4 节), 这些工具允许您在 \TeX 中使用操作系统的字体并无需安装。(但请注意, 使用系统字体通常会使您的文档源文件在其他人的不同环境中无法使用。)

3.5 测试安装是否成功

在完成 \TeX Live 安装之后, 自然你会希望试试看它是否正常工作, 好让你在以后能够创建优美的文档和字体。

你可能马上需要的是一个用来编辑文件的前端程序。 \TeX Live 只在 Windows 下安装了 \TeX works (<https://tug.org/texworks>), 而 Mac \TeX 下则安装了 \TeX Shop (<https://pages.uoregon.edu/koch/texshop>)。在其他 Unix 系统下, 则由你自己选择编辑器。当然, 存在很多选择, 有些在下一小节列出了; 还可以参见 <https://tug.org/interest.html#editors>。任何纯文本编辑器都可以, 并不需要专门为 \TeX 设计。

这个小节后面给出了一些测试系统是否正常工作的基本步骤。我们这里使用的是 Unix 命令, 在 macOS 和 Windows 下, 你更可能使用图形界面运行这些测试, 不过其原理并无不同。

1. 首先确认你可以执行 `tex` 程序:

```
> tex --version
```



```
TeX 3.14159265 (TeX Live ...)
Copyright ... D.E. Knuth.
...
```

如果返回的结果是 ‘command not found’ 而非版本和版权信息，或者显示了旧版本的信息，很有可能是因为你没有把正确的 bin 子目录添加到 PATH 中。参见第 18 页关于设置环境变量的说明。

2. 处理一个基本的 L^AT_EX 文件:

```
> pdflatex sample2e.tex
This is pdfTeX 3.14...
...
Output written on sample2e.pdf (3 pages, 142120 bytes).
Transcript written on sample2e.log.
```

如果无法找到 sample2e.tex 或其他什么文件，很可能是因为旧的环境变量或配置文件影响了判断。我们建议你重置所有 T_EX 相关的环境变量然后重试。(你可以让 T_EX 报告具体搜索的路径，以便仔细分析出错的原因。参见第 38 页的“调试操作”一节以了解更多信息。)

3. 即时预览结果:

```
> xpdf sample2e.pdf
```

你应该可以看到在新窗口中出现了一篇介绍 L^AT_EX 基础的有趣文档。(如果你刚接触 T_EX 系统，还是值得一读的。)

当然还有许多其他的 PDF 阅读器。在 Unix 系统, evince 和 okular 是常见的选择。在 Windows 系统, 我们建议尝试 Sumatra PDF (<https://www.sumatrapdfreader.org/free-pdf-reader.html>)。T_EX Live 中并没有包含任何 PDF 阅读器，因此，您必须单独安装您想要使用的任意一款。

4. 当然你任何可以生成 T_EX 原始的 DVI 文件:

```
> latex sample2e.tex
```

5. 并且在线预览 DVI

```
> xdvi sample2e.dvi # Unix
> dviout sample2e.dvi # Windows
```

xdvi 需要运行在 X 窗口系统下才能工作; 如果没有运行窗口环境或 DISPLAY 环境变量设置错误, 都会得到 ‘Can't open display’ 这句错误信息。

6. 由 DVI 创建 PostScript 文件:

```
> dvips sample2e.dvi -o sample2e.ps
```

7. 或由 DVI 创建 PDF 文件，这是一种使用 pdfT_EX (或 XeT_EX 或 LuaT_EX) 的替代方案，它有时候会很有用:

```
> dvipdfmx sample2e.dvi -o sample2e.pdf
```

8. 除 sample2e.tex 外可能会对你有用的其他标准测试文件:

small2e.tex 比 sample2e 更为简单的文档，供你在遇到问题时尝试减少输入的内容。

testpage.tex 测试你的打印机是否带有预设的偏移量。

nfssfont.tex 打印一份字体表格以供测试。

testfont.tex 用 plain T_EX 打印字体表格。

story.tex 最经典的 (plain) T_EX 测试文件。在执行 ‘tex story.tex’ 之后，你还要在 * 提示符下键入 ‘\bye’。

9. 如果你安装了 xetex 包，可以按如下步骤测试它能否访问系统字体:

```
> xetex opentype-info.tex
This is XeTeX, Version 3.14...
...
```

Output written on opentype-info.pdf (1 page).
 Transcript written on opentype-info.log.

如果你收到 “Invalid fontname ‘Latin Modern Roman/ICU’...” 这样的错误信息，就说明需要配置系统才能找到 T_EX Live 自带的字体。参见第 3.4.4 节。

3.6 卸载 T_EX Live

要卸载 T_EX Live (在成功安装后; 关于 Windows 的操作, 请见最后):

```
tlmgr uninstall --all
```

系统会请求确认, 否则不会执行任何操作。(如果没有使用 `--all`, `uninstall` 操作将用于移除单个包。)

这不会移除用户特定的目录, 具体如下 (也参见第 2.3 节):

TEXMFCONFIG 用于用户配置更改。如果你希望保留它们, 请确保在删除之前知道如何重新创建它们。
TEXMFVAR 用于存储自动生成的缓存数据, 例如本地格式文件。除非你将其用于其他目的, 否则删除它应该是安全的。

TEXMFHOME 只包含你自己安装的文件, 通常是发行版中不提供的文件。除非你完全停止使用 T_EX, 或者想从头开始, 否则你可能不希望删除此目录。

您可以通过运行 `kpsewhich -var-value=var` 来找到这些变量的目录路径。

此 `tlmgr` 卸载操作也不会撤销安装后的操作, 如在你的 `shell` 初始化文件中修改的 `PATH` 和系统对 T_EX Live 中字体的访问 (见第 3.4 节)。如果需要, 您必须手动撤销这些操作。

在 Windows 上, 卸载可以通过 GUI 进行; 见第 6.1 节。

3.7 其他可下载软件的链接

如果你还是个 T_EX 新手, 或者在编辑 T_EX 或 L^AT_EX 文档时需要帮助, 请访问 <https://tug.org/begin.html> 寻找引导性的资源。

这里是一些你可能会考虑安装的其他工具的链接。

Ghostscript <https://ghostscript.com/> 一个免费的 PostScript 和 PDF 解释器。

Perl <https://perl.org/> 与 CPAN 中的补充包, <https://cpan.org/>。

ImageMagick <https://imagemagick.org>, 用于图形处理和转换。

NetPBM <http://netpbm.sourceforge.net>, 同样用于图形。

面向 T_EX 的编辑器 有很广泛的选择, 一般依用户个人的口味而定。这里按字典序列出了一些 (部分是 Windows 才有的)。

- GNU Emacs 在所有的主要平台上都可以使用; 参见 <https://www.gnu.org/software/emacs>。
- AUCT_EX 在 Emacs 下运行; 通过 Emacs 的包管理器 ELPA 可以获得。源代码也可以通过 CTAN 获取。AUCT_EX 的主页是 <https://www.gnu.org/software/auctex>。
- SciTE 在 <https://www.scintilla.org/SciTE.html> 提供。
- Texmaker 是自由软件, 在 <https://www.xmlmath.net/texmaker> 提供。
- TeXstudio 是 Texmaker 的一个 fork, 引入了额外的功能; 在 <https://texstudio.org> 提供。
- TeXnicCenter 是自由软件, 在 <https://www.texniccenter.org> 提供。

- TeXworks 是自由软件，在 <https://tug.org/texworks> 提供，也作为 T_EX Live 的一部分只在 Windows 下被安装。
- Vim 是自由软件，在 <https://www.vim.org> 提供。
- WinEdt 是共享软件，在 <https://tug.org/winedt> 或 <https://www.winedt.com> 提供。
- WinShell 在 <https://www.winshell.de> 提供。

关于这类软件包和程序，<https://tug.org/interest.html> 有一份更长的列表。

4 特殊安装

前面的章节描述了基本的安装过程。这里我们介绍一些特殊的情形。

4.1 用户共享 (或跨机器) 安装

T_EX Live 的设计可以使它在同一个网络中的不同的系统间共享。在标准的目录结构下，不需要配置固定的绝对路径：T_EX Live 程序所需要的文件都能通过这些程序自身的相对路径找到。你可以在 \$TEXMFDIST/web2c/texmf.cnf 配置文件中看到实际的处理，它包含了类似下面的内容：

```
TEXMFROOT = $SELFAUTOPARENT
...
TEXMFDIST = $TEXMFROOT/texmf-dist
...
TEXMFLOCAL = $SELFAUTOGRANDPARENT/./texmf-local
```

这就意味着，其它的系统或用户只需要把 T_EX Live 的可执行文件的位置添加到其系统的搜索路径中就可以使用了。

同理，你也可以先把 T_EX Live 安装在本地，然后再把整个安装目录转移到网络上。

至于 Windows，T_EX Live 包含了一个叫 tlaunch 的启动程序。它的主窗口的菜单和按钮包括许多 T_EX 相关功能的程序和文档，还可以通过 ini 定制。首次使用时，它会进行一些 Windows 专有的后安装设置，比如给当前用户修改搜索路径并创建文件关联。因此，在局域网中可以访问到 T_EX Live 的工作站，只需要启动器的快捷菜单就够了。参见 tlaunch 手册 (texdoc tlaunch, 或 <https://ctan.org/pkg/tlaunch>)。

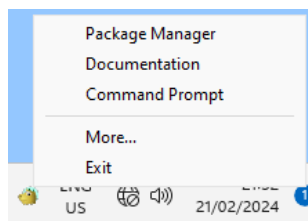
4.2 便携 (USB) 安装

-portable 安装程序选项 (或者文本安装程序的 v 命令，或对应的 GUI 选项) 创建的是一套在一个独立路径下完全自包含的 T_EX Live 安装，而跳过系统集成。你可以在 USB 盘上创建这样的安装，或者在安装后复制到 USB 盘上。

从技术上讲，通过将 TEXMFHOME、TEXMFVAR 和 TEXMFCONFIG 的默认值分别设置为与 TEXMFLOCAL、TEXMFSYSVAR 和 TEXMFSYSCONFIG 相同，使得便携式安装成为自包含的；因此，不会创建针对用户的配置和缓存。

要使用这样的便携安装来运行 T_EX，你必须将对应的二进制目录加入终端的搜索路径，如往常一样。

在 Windows 下，你可以双击安装根目录下的 `tl-tray-menu` 来创建一个临时的启动栏菜单，以选择执行一些常见任务，如这个截图所示：



‘More...’ 项目解释了如何定制这个菜单。

5 tlmgr: 管理你的安装

T_EX Live 包含一个叫 `tlmgr` 的程序，它可以用来管理安装后的系统。它的功能包括：

- 安装、更新、备份、还原和卸载各个软件包，可选择考虑依赖关系；
- 查找和列出软件包以及它们的描述；
- 列出、添加和删除不同平台的可执行文件；
- 改变安装选项，比如纸张大小和源文件位置（参见第 3.3.1 节）。

`tlmgr` 的功能已经完全覆盖了 `texconfig` 原来的功能。考虑到有人已经习惯了原有的界面，我们仍然发行和维护 `texconfig`，现在但我们建议使用 `tlmgr`。

5.1 tlmgr 的 GUI 界面

T_EX Live 包含多个 `tlmgr` 的 GUI 前端。其中两个值得注意的：图 9 展示了 `tlshell`，它是用 Tcl/Tk 编写的，在 Windows 开箱即用。图 10 展示了 `tlcockpit`，它需要 Java 版本 8 或者更高以及 JavaFX。这两个都是独立的安装包。

`tlmgr` 还有一个原生的 GUI 模式（见图 11），启动它可以用：

```
> tlmgr -gui
```

但是 GUI 扩展需要 Perl/Tk，而这个模块已经不再包含在 T_EX Live 的 Windows Perl 分发中了。

5.2 tlmgr 命令行使用示例

在初始安装之后，你可以用下面的命令更新至最新版本：

```
> tlmgr update -all
```

如果这太激进了一点，先尝试：

```
> tlmgr update -all -dry-run
```

或（产生更少输出）：

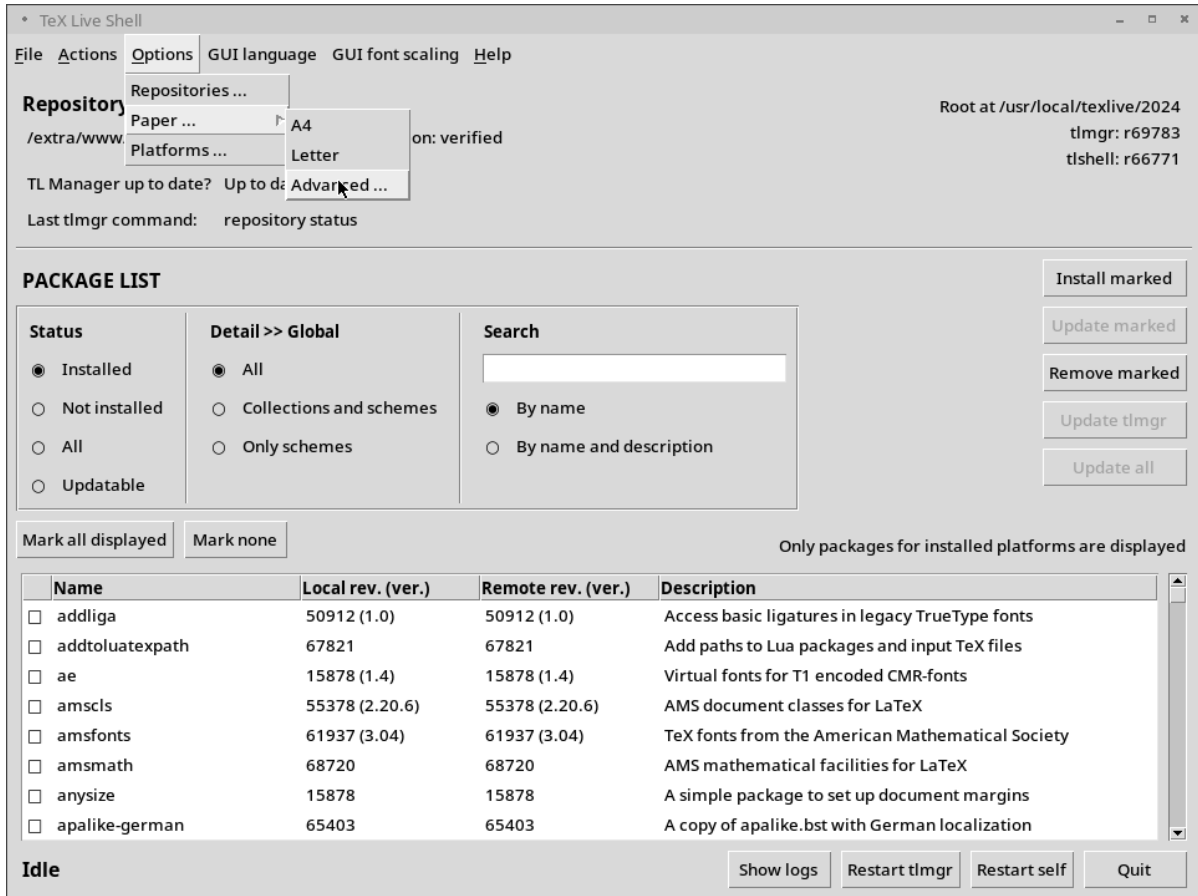


图 9: tlmgr GUI, 显示 Actions 菜单 (GNU/Linux)

```
> tlmgr update -list
```

下面这个更复杂一点的例子从本地目录添加了一个新的软件包集合, 用于 XeTeX 引擎:

```
> tlmgr -repository /local/mirror/tlnet install collection-xetex
```

它会产生下面的输出 (节略部分):

```
install: collection-xetex
install: arabxetex
...
install: xetex
install: xetexconfig
install: xetex.i386-linux
running post install action for xetex
install: xetex-def
...
running mktexlsr
mktexlsr: Updating /usr/local/texlive/2024/texmf-dist/ls-R...
```

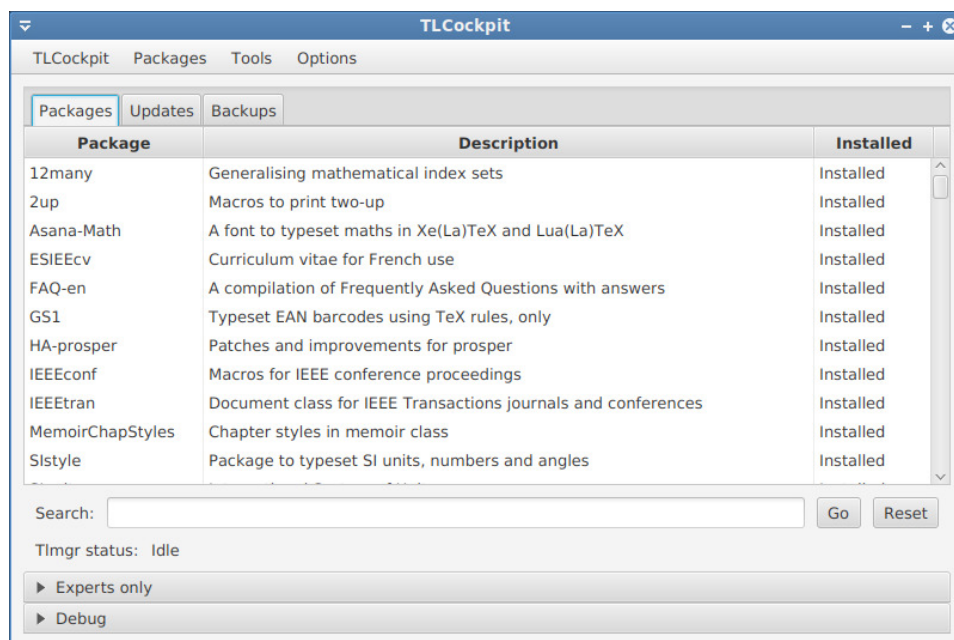


图 10: tlmgr 的 tlcocpit GUI

```
...
running fmtutil-sys --missing
...
Transcript written on xelatex.log.
fmtutil: /usr/local/texlive/2025/texmf-var/web2c/xetex/xelatex.fmt installed.
```

如你所见，tlmgr 会安装所有依赖的包，也会处理所有包括刷新文件名数据库和重新生成格式文件在内的所有必要的后安装工作。上面给 XeTeX 生成了新的格式文件。

要描述一个包 (或者集合、安装方案):

```
> tlmgr show collection-latexextra
```

会输出这样的内容:

```
package:    collection-latexextra
category:   Collection
shortdesc:  LaTeX supplementary packages
longdesc:   A very large collection of add-on packages for LaTeX.
installed:  Yes
revision:   46963
sizes:      657941k
```

最后也是最重要的，查阅 <https://tug.org/texlive/tlmgr.html> 这里的完整文档，或者:

```
> tlmgr -help
```

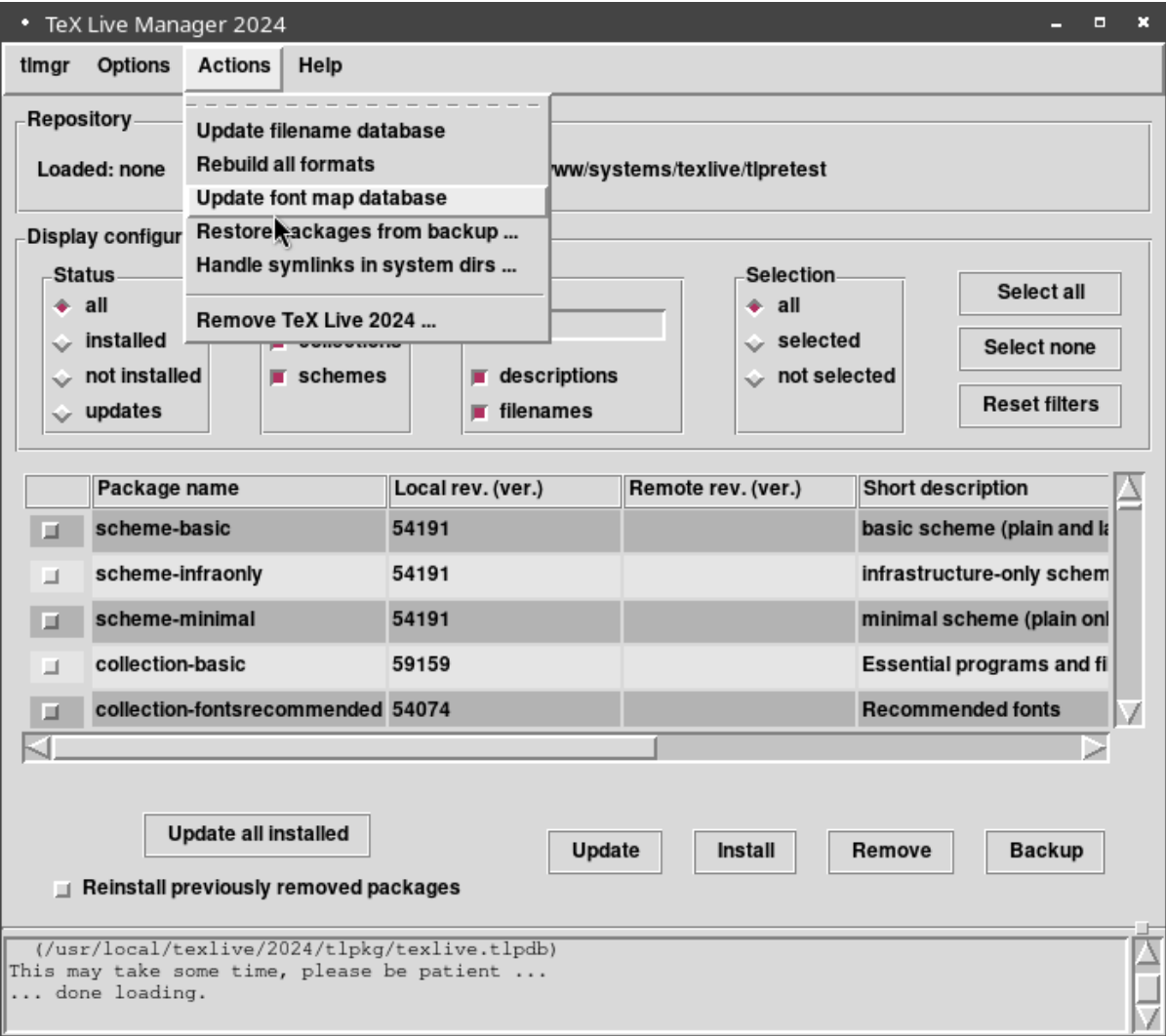


图 11: 旧有的 tlmgr GUI 模式: 按下 ‘Load’ 后的主窗口

6 有关 Windows 平台的说明

6.1 针对 Windows 的特征

在 Windows 下，安装程序执行了一些额外的操作:

菜单与快捷方式。 在开始菜单上加入了新的 ‘TeX Live’ 程序菜单，主要是一些 GUI 程序，如 tlmgr 的 GUI 和 dviout, 以及一些文档的菜单。

文件关联。 如果启用，TeXworks 和 Dviout 会成为它们对应文件类型的默认程序，或者在这些文件类型的 “用... 打开” 右键菜单中出现。然而，优先级更高的 “用户选择” 文件关联 (只能通过交互方式

指定) 可能会造成干扰。

PostScript 支持。 对于 PostScript 文件, 一个 PSviewer 文件类型现在会将 PostScript 转换为临时 PDF, 然后由默认的 PDF 查看器显示。各种位图格式在其“打开方式”右键菜单中会有一个 bitmap2eps 条目, 用于转换为 EPS, 让 sam2p 或 bmeps 执行实际工作。

自动路径调整。 不需要手动配置步骤。

卸载程序。 安装程序会为 T_EX Live 创建一个条目, 要么在“添加/删除程序”中 (管理员安装), 要么在 T_EX Live 菜单中 (单用户安装)。

写保护。 对于管理员安装, 如果 T_EX Live 安装在正常的 NTFS 格式化的不可移动磁盘上, 则 T_EX Live 目录是写保护的。

另外, 另一种方式可以参见 tlaunch, 在第 4.1 节有描述。

6.2 Windows 上附加的软件

为了使安装更加完整, T_EX Live 需要支持那些 Windows 机器上不常见的软件包。T_EX Live 提供了以下缺失的部分。这些程序只在 T_EX Live 的 Windows 版本中提供。

Perl、Tcl/Tk 和 Ghostscript。 由于 Perl 和 Ghostscript 的重要性, 并且安装程序和 t_lshell GUI 是用 Tcl/Tk 编写的, T_EX Live 提供了这些程序的‘隐藏’拷贝。需要这些软件支持的 T_EX Live 程序知道它们的位置, 但它们不会通过环境变量和注册表设置来暴露所在的位置。它们不是完整的安装版 (除了 Ghostscript), 也不会与任何 Perl、Tcl/Tk 或 Ghostscript 系统安装程序冲突。请参阅第 6.3 节, 了解如何告知 T_EX Live 使用您自己的外部安装来处理 T_EX Live 内的贡献脚本。

dviout。 另外安装的还包括一款 DVI 预览程序 dviout。当你第一次使用 dviout 预览文件时, 因为没有安装屏幕显示字体, 它将生成字体。一段时间后, 你所使用的大部分字体都将生成, 随后, 你将很少再看到生成字体的窗口。你可以从 (强烈推荐的) 在线帮助中获得更加详细的信息。

T_EXworks。 T_EXworks 是一个集成了 PDF 阅读器的 T_EX 编辑器。

命令行工具。 与常见的 T_EX Live 二进制文件一起, 还安装了一些常见的 Unix 命令行工具的 Windows 移植版本。它们包括 gzip、zip、unzip 以及 poppler 套件中的工具 (如 pdffinfo、pdffonts 等)。

fc-list、fc-cache 等。 来自于 fontconfig 库的这些工具有助于 XeT_EX 处理 Windows 的系统字体。你可以使用 fc-list 来确定传递给经 XeT_EX 扩展后的 \font 命令的字体名称。如果需要, 首先运行 fc-cache 更新字体信息。

6.3 使用外部的 Perl、Tcl/Tk 和 Ghostscript 程序

通常, T_EX Live 会使用其内置的 Perl、Tcl/Tk 和 Ghostscript 来运行 T_EX Live 内部的贡献脚本。如果你希望使用自己的外部版本, 你可以在安装根目录的 texmf.cnf 文件中进行配置。

对于 Perl, 你需要添加一行

```
TEXLIVE_WINDOWS_TRY_EXTERNAL_PERL = 1
```

T_EX Live (特别是 bin/windows/runscript.tlu) 会在搜索路径中查找 perl.exe, 除了属于 T_EX Live 底层的脚本之外。这是三者中最有可能有用的设置; 尽管 T_EX Live Perl 包含许多附加模块, 但它不能支持所有的第三方脚本。

同样地, 对于 Tcl/Tk, 你需要添加一行

```
TEXLIVE_WINDOWS_TRY_EXTERNAL_TCL = 1
```

T_EX Live 将会在搜索路径中查找 `tclkit.exe`、`wish.exe`、`wish85.exe`、`wish86.exe` 和 `wish87.exe`。

Ghostscript 的处理方式有所不同，你需要指定命令行 Ghostscript 的文件名或完整路径：

`TEXLIVE_WINDOWS_EXTERNAL_GS` = 命令行 Ghostscript 的路径

另一个不同点是，T_EX Live 附带的 Ghostscript 是完整的，仅缺少文档和打印机驱动程序。因此，你不太可能需要覆盖它。

另请参阅第 7.1.2 节关于 `texmf.cnf` 的说明。

6.4 User Profile 目录相当于主目录

Windows 下对应于 Unix 下的主目录的是 `%USERPROFILE%`。在 Windows Vista 和以后的系统下，它通常位于 `C:\Documents and Settings\<username>`；在 Windows Vista 和之后的版本下是 `C:\Users\<username>`。通常情况下，在 `texmf.cnf` 文件和 Kpathsea 中，`~` 在 Windows 和 Unix 下均可以进行合适的展开。

6.5 Windows 注册表

Windows 将几乎所有的配置数据存储在其注册表中。注册表包含一组层次结构组织的键，具有多个根键。对于安装程序来说，最重要的根键是 `HKEY_CURRENT_USER` 和 `HKEY_LOCAL_MACHINE`，简称为 `HKCU` 和 `HKLM`。注册表的 `HKCU` 部分位于用户的主目录中（参见第 6.4 节）。`HKLM` 通常位于 Windows 目录的一个子目录中。

在某些情况下，系统信息可以从环境变量中获取，但对于其他信息，例如快捷方式的位置，则需要查询注册表。永久设置环境变量也需要访问注册表。

6.6 Windows 权限

在较新的 Windows 版本中，普通用户与管理员是有区别的。只有管理员能自由访问近乎整个操作系统。我们努力使得 T_EX Live 的安装不需要管理员权限。

如果安装程序以管理员权限启动，会有选项允许给所有用户安装，如果启用了这个选项，就会给所有用户创建快捷方式，并修改系统搜索路径。否则，安装程序只为当前用户创建快捷方式，并改变用户搜索路径。

无论管理员状态如何，T_EX Live 预设的默认根目录总是位于 `%SystemDrive%` 下。对于当前用户安装，程序总是要测试根目录是否可写。

如果用户不是管理员，并且 T_EX 已经存在搜索路径中，则可能会出现这个问题。因为有效搜索路径是由系统路径后接用户路径组成，新安装的 T_EX Live 可能永远不会优先运行。为了保险起见，安装程序为命令提示符创建快捷方式。在这个快捷方式中，新安装的 T_EX Live 的可执行目录被预设于本地的搜索路径中。当从这个快捷方式启动命令行任务时，便可以使用新的 T_EX Live。如果安装了 T_EXworks，其快捷方式也将 T_EX Live 加进了搜索路径中，所以它应该不会出现路径问题。

你需要知道的是：即使你以管理员身份登录系统，依旧要求你提供管理员权限。实际上，是否以管理员身份登录并不是问题所在。相反，在你希望运行的程序或快捷方式上单击右键，系统通常会给出“以管理员身份执行”这样的选择。

6.7 增加 Windows 和 Cygwin 下的最大内存量

Windows 和 Cygwin (参见第 3.1.4 节以了解 Cygwin 安装细节) 的用户可能会在执行特定 T_EX Live 程序时遇到内存不足的情况。例如 asy 在你尝试分配一个 25,000,000 个实数的数组时可能会内存不足, 而 LuaT_EX 在你尝试处理一个包含大量字体的文档时可能会内存不足。

对 Cygwin, 你可以按照 Cygwin 用户指南 (<https://cygwin.com/cygwin-ug-net/setup-maxmem.html>) 中的办法增加可用内存量。

对 Windows, 你必须创建一个文件, 例如命名为 moremem.reg, 包含如下内容:

```
Windows Registry Editor Version 5.00
```

```
[HKEY_LOCAL_MACHINE\Software\Cygwin]
"heap_chunk_in_mb"=dword:ffffff00
```

然后以管理员身份执行命令 `regedit /s moremem.reg`。(如果你只希望修改当前用户而不是整个系统的内存, 改用 HKEY_CURRENT_USER。)

7 Web2C 用户指南

Web2C 是一整套 T_EX 相关程序的集合: T_EX 本身、METAFONT、MetaPost、BibT_EX, 等等。它是 T_EX Live 的核心。Web2C 的主页及最新的手册等都在 <https://tug.org/web2c>。

我们简单的介绍一下它的历史: 最早它是由 Tomas Rokicki 在 1987 年实现的, 他开发了第一套将 T_EX 系统的代码转换为 C 语言代码的系统, 基于的是 Unix 下 change files 的原理, change files 的工作是 Howard Trickey 和 Pavel Curtis 完成的。Tim Morgan 后来成为了这套系统的维护者, 在这期间, 软件的名称改为了 Web-to-C。在许多其他贡献者的帮助下, 1990 年 Karl Berry 接手了这个工作, 到 1997 年, 他把这项工作交给了 Olaf Weber。Olaf Weber 在 2006 年又把这项工作交还给了 Karl。

Web2C 系统可以在 Unix、Windows 系统、macOS 和其他的一些操作系统下运行。它使用的是 Knuth 用 WEB 文学编程语言编写的 T_EX 和其他基本程序的原始代码, 将其转换为 C 源码。用这种方法处理的核心 T_EX 程序包括:

- bibtex 维护参考文献。
- dvicopy 展开 DVI 中的虚拟字体 (virtual font) 引用。
- dvitomp 将 DVI 转换为 MPX (MetaPost 图片)。
- dvitype 将 DVI 转换为可读文本。
- gftodvi 生成 Generic 格式字体的 proofsheets。
- gftopk 将 Generic 格式字体转换为 packed 格式字体。
- gftype 将 Generic 格式字体转换为可读文本。
- mf 创建字体。
- mft 以漂亮的方式排版输出 METAFONT 的代码。
- mpost 创建技术性插图。
- patgen 创建断字规则文件。
- pktoif 将 Packed 格式字体转换为 generic 格式字体。
- pktype 将 PK 格式转换为可读的文本。
- pltotf 将纯文本的 property list 转换为 TFM 格式。
- pooltype 显示 WEB 的 pool 文件。

`tangle` 将 WEB 转换为 Pascal 代码。
`tex` 排版。
`tftopl` 将 TFM 格式转换为纯文本的 property list 格式。
`vftovp` 将虚拟字体格式转换为 virtual property list 格式。
`vptovf` 将 virtual property list 格式转换为虚拟字体格式。
`weave` 将 WEB 转换为 $\text{T}_{\text{E}}\text{X}$ 。

这些程序的详细功能和调用语法都在其各自软件包的文档中有说明，在 Web2C 的文档中也有相关介绍。不过，有些规则对所有这些程序都是通用的，了解这些规则有助于你更好的使用 Web2C。

所有的程序都接受这些 GNU 标准的选项：

`--help` 显示基本使用说明。
`--version` 显示版本信息，然后退出。

而大部分都接受：

`--verbose` 显示详细的执行过程。

所有的 Web2C 程序均使用 Kpathsea (<https://tug.org/kpathsea>) 路径搜索库来查找文件，这套库结合环境变量和配置文件的使用来优化 (庞大的) $\text{T}_{\text{E}}\text{X}$ 文件的搜索。Web2C 可以在多于一套的目录树下执行查找，这可以方便维护类似 $\text{T}_{\text{E}}\text{X}$ 标准发行版和本地版本的扩展这样两套目录树。为了优化搜索的速度，每个目录树的顶层目录下都有一个 `ls-R` 文件，这个文件里包含了所有此目录下文件的名称和对应的相对路径。

7.1 Kpathsea 路径搜索

我们首先介绍一下 Kpathsea 库的通用路径搜索方式。

我们将目录名称称作路径元素，而把用冒号或者分号分隔的路径元素列表称作搜索路径。搜索路径可能是许多种来源的组合，比如在 `./dir` 路径下搜索 `my-file` 这个文件，Kpathsea 将逐个尝试路径中的每个元素：首先是 `./my-file`，然后是 `/dir/my-file`，并返回找到的第一个结果 (或者也可以返回所有的结果)。

为了符合所有操作系统下的习惯，Kpathsea 在非 Unix 系统下使用的路径分隔符可能不是冒号 (`:`) 和斜杠 (`/`)。

在检查一个具体的路径元素 p 时，Kpathsea 首先检查是否有符合 p 的文件名数据库 (见第 35 页的“文件名数据库”) 存在，也就是说，是否有数据库正好对应着 p 的一个前缀。如果存在，就在数据库中寻找符合的路径后缀。

尽管最简单也最常见的路径元素是目录名称，Kpathsea 搜索的路径里还可以使用其他额外功能：多层默认值，环境变量名称、配置文件值、用户主目录，以及递归式子目录查找。所以我们把 Kpathsea 将搜索路径变换为一个或多个基本目录名的过程称为展开路径元素的过程。展开的方式按执行的顺序在后续小节里有叙述。

注意，如果搜索的文件给出了绝对路径或者明确的相对路径，即以 `/` 或 `./` 或 `../` 起始，Kpathsea 将只检查该文件是否存在。

7.1.1 路径的来源

搜索路径可能来自许多地方，Kpathsea 是按照下面的顺序查找的：

1. 用户设置的环境变量，例如 TEXINPUTS。以 连接某个程序名称的环境变量有更高的优先级，比如若正在运行的程序是 ‘latex’，那 TEXINPUTS.latex 将比 TEXINPUTS 优先级更高。
2. 专门针对某个程序的配置文件，比如 dvips 的 config.ps 里出现 ‘S /a:/b’ 这样一行。
3. Kpathsea 配置文件 texmf.cnf，包含类似 ‘TEXINPUTS=/c:/d’ 这样的一行（参见下面的解释）。
4. 编译时的缺省值。

你可以通过调试选项看到所有的这些值（参见第 38 页的“调试操作”）。

7.1.2 配置文件

Kpathsea 读入运行时配置文件 texmf.cnf 来获得搜索路径和其他定义。我们用 TEXMFCNF 搜索路径来查找这些文件，但我们不建议设置这个（或者任何）环境变量来替代系统路径。

作为代替，标准的安装将产生文件 .../2025/texmf.cnf。如果你必须修改缺省值（一般不需要），这是修改的地方。主配置文件是 .../2025/texmf-dist/web2c/texmf.cnf。你不应该修改后者，因为在发行版本被更新后会丢失修改。

此外，如果只希望把一个个个人目录加入特定的搜索路径，设置环境变量是合理的：

```
TEXINPUTS=./my/macro/dir:
```

为保证这个设计可以一直被维护和兼容，在末尾加入一个 ‘.’（Windows 下用 ‘;’）来加入系统路径，而不是完全替代系统路径（参见第 7.1.4 节）。另一个选项是使用 TEXMFHOME 树（参见第 3.2.3 节）。

搜索路径里所有的 texmf.cnf 文件都会被读入，而先读入的优先级更高。比如搜索路径是 .:\$TEXMF，那么文件 ./texmf.cnf 里面的值要比 \$TEXMF/texmf.cnf 里边的优先。

- 以 %——要么在行首，要么在空白后面——表示单行注释。
- 忽略空行。
- 行末的 \ 作为连接符，即把下一行直接接上。但保留下一行行首的空白。
- 所有剩余的行格式如下：

```
variable [.progrname] [=] value
```

 ‘=’ 号和空白都是可选的。（但如果 value 以 ‘.’ 起始，最好的办法是用 ‘=’ 来避免 ‘.’ 被识别为程序名称。）
- variable（变量）允许包含任何字符，除空白、‘=’、‘.’ 之外。不过只用 ‘A-Za-z_’ 是最保险的。
- 如果 ‘.progrname’（程序名）存在，则该定义只对正在运行的名叫 progrname 或 progrname.exe 的程序起作用。这可以让给不同种类的 T_EX 程序设置不同的搜索路径。
- 就像字符串一样，value 可以包含任何字符。但是，实际上大部分 texmf.cnf 的值都和路径展开有关，而因为展开会用到多种特殊字符（参见第 7.1.7 节），比如大括号和逗号，所以它们不能用作路径名称。
 在 Unix 下 value 里的 ‘;’ 会被转换为 ‘.’，这样同一份 texmf.cnf 可以同时支持 Unix 和 Windows 系统。这个转换适用于任何值，不仅仅是搜索路径，不过实际上 ‘;’ 也不会其他值里用到。
 \$var.prog 的功能在右侧无效，你必须多用一个变量才行。
- 在读入所有定义后再开始展开，所以你可以引用后边才定义的变量。

展示上面所有内容的一段配置文件如下：

```

TEXMF          = {$TEXMFLOCAL,!!$TEXMFMAIN}
TEXINPUTS.latex = .;$TEXMF/tex/{latex,generic;}//
TEXINPUTS.fontinst = .;$TEXMF/tex//;$TEXMF/fonts/afm//
% e-TeX related files
TEXINPUTS.elatex = .;$TEXMF/{etex,tex}/{latex,generic;}//
TEXINPUTS.etex   = .;$TEXMF/{etex,tex}/{eplain,plain,generic;}//

```

7.1.3 路径展开

和 Unix shell 类似, Kpathsea 能够识别搜索路径中的特殊字符。比如路径 `~$USER/{foo,bar}//baz`, 将展开为这样的子目录: 在 `$USER` 的主目录下的 `foo` 或 `bar` 目录中, 且包含 `baz` 文件或目录。这种展开将在下面解释。

7.1.4 默认展开

如果最高优先级 (参见第 33 页的“路径来源”) 的搜索路径中包含一个额外的冒号 (即前置、后置或连续的冒号), Kpathsea 将在此处插入次高优先级的搜索路径。如果插入的那个路径里也有额外的冒号, 同样的步骤将发生在更次以及优先级的路径上。假设环境变量设置为

```
> setenv TEXINPUTS /home/karl:
```

而 `texmf.cnf` 里的 `TEXINPUTS` 值为

```
.: $TEXMF//tex
```

则用于搜索的最终值为:

```
/home/karl:.: $TEXMF//tex
```

因为没必要插入多个相同的值, 所以 Kpathsea 只会修改一个额外的 ‘:’, 其他的不变。它首先检查前置的 ‘:’, 然后是末尾的 ‘:’, 最后是连续的 ‘:’。

7.1.5 大括号展开

大括号展开是一项有用的特性, 其作用是把例如 `v{a,b}w` 这样的转换为 `vaw:vbw`, 允许嵌套使用。通过把 `$TEXMF` 赋值为一个括号列表, 可以构造出多套 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 层级结构。在分发的 `texmf.cnf` 里有下面的定义 (简化的例子):

```
TEXMF = {$TEXMFVAR,$TEXMFHOME,!!$TEXMFLOCAL,!!$TEXMFDIST}
```

比如我们可以用它来设置 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 的输入路径:

```
TEXINPUTS = .;$TEXMF/tex//
```

的时候, 检查完当前目录后, 依次检查的路径是 `$TEXMFVAR/tex`, `$TEXMFHOME/tex`, `$TEXMFLOCAL/tex` 和 `$TEXMFDIST/tex` (后两个只在 `ls-R` 数据库中搜索)。

7.1.6 子目录展开

在路径元素里的目录名称 d 后面接连使用两个或更多连续的斜杠，表示 d 的所有子目录：首先是直接处于 d 下的那些，然后是这些子目录的子目录，依此类推。每层的目录出现的顺序是不一定的。

如果你在 ‘//’ 后面还指定了文件名，匹配的将只是那些包含了指定文件的路径。比如 ‘/a//b’ 将展开为路径 /a/1/b, /a/2/b, /a/1/1/b 等等，但不会展开为 /a/b/c 或 /a/1。

可以在单个路径元素里使用多个 ‘//’，但出现在路径开头的 ‘//’ 将被忽略。

7.1.7 texmf.cnf 文件中的特殊字符简要说明

下面的列表总结了 Kpathsea 配置文件中出现的特殊字符和构成：

```

:      路径分隔符，在路径的前边或者末尾，或者在中间重复出现时，表示默认的展开方式。
;      非 Unix 系统下的路径分隔符 (和 : 功能一样)。
$      变量展开。
~      表示用户的个人主目录。
{...}  大括号展开。
,      在大括号展开时分隔项目。
//     子目录展开 (可以出现在除路径开头外的任意位置)。
% and # 注释的起始。
\      在行末连接下一行的字符，以支持跨行的设置项。
!!     只在数据库中搜索文件，不搜寻磁盘。

```

具体哪一个字符会被视为特殊字符还是不会，取决于它使用的上下文。这些规则在配置文件的多层的解析过程中都会体现 (包括解析、展开、搜索，等等)，所以不能简短地说明。也没有通用的转义方法，尤其 ‘\’ 不是 texmf.cnf 文件中的“转义字符”。

在选择安装使用的目录名时，避免使用这些字符里的任何一个最为保险。

7.2 文件名数据库

Kpathsea 采取了一些措施来尽量减少搜索时的磁盘访问。尽管如此，在标准的 T_EX Live 中，或者在任何包含足够多目录的安装中，为查找某个文件而搜索每一个可能的目录将会耗费过长的时间。因此，Kpathsea 可以使用一个名为 ls-R 的外部构建的纯文本“数据库”文件，该文件将文件映射到目录，从而避免了对磁盘进行全面搜索的必要。

第二个数据库 aliases 允许你给 ls-R 中的文件指定其他的名字。

7.2.1 文件名数据库

如上所述，主文件名数据库的名称必须是 ls-R。你可以在安装路径下每个希望被搜索的 T_EX 层次结构的根目录中放置一个 ls-R 文件 (默认情况下为 \$TEXMF)。Kpathsea 会沿着 TEXMFDBS 路径查找 ls-R 文件。

推荐使用发行版中包含的 mktexlsr 脚本来创建和维护 ‘ls-R’ 文件。脚本由各种 ‘mktex’... 脚本调用。原则上，这个脚本只是执行下面的命令：

```
cd /your/texmf/root && \ls -lLAR ./ >ls-R
```

意味着你的系统的 `ls` 命令生成正确的输出格式 (GNU `ls` 是没问题的)。要保证数据库及时更新, 最简单的方法是定期通过 `cron` 来重建。这样当安装的文件发生变化时 (例如安装或更新 L^AT_EX 包之后), 它会自动更新。

如果在数据库找不到文件, 缺省情况下 Kpathsea 会继续搜索磁盘。然而, 如果某个特定的路径元素以 ‘!’ 开头, 则只会在针对这一元素的数据库中查找而不去搜索磁盘。

7.2.2 kpsewhich: 独立的路径搜索

`kpsewhich` 程序独立于任何特定应用程序进行路径搜索。它可以作为类似 `find` 这样的程序, 专门在 T_EX 层级结构中定位文件 (这在发行版中的 ‘`mktex`’... 脚本中使用得非常多)。

```
> kpsewhich option... filename...
```

在 *option* 中指定的选项以 ‘-’ 或 ‘--’ 开头, 且任何不含歧义的缩写都是可以接受的。

Kpathsea 将命令行中的每个非选项参数视为文件名进行查找, 并返回找到的第一个文件。它不提供可以返回具有特定名称的所有文件的选项 (可以使用 Unix 的 ‘`find`’ 工具来实现此功能)。

接下来将介绍最常用的选项。

- `--dpi=num` 将分辨率设置为 *num*; 这仅影响 ‘gf’ 和 ‘pk’ 的查找。为了兼容 `dvips`, ‘-D’ 是一个同义词。默认值是 600。
- `--format=name`
将查找的格式设置为 *name*。默认情况下, 格式是从文件名中猜测的。对于没有相关明确后缀的格式, 例如 MetaPost 支持文件和 `dvips` 配置文件, 你必须指定 Kpathsea 已知的名称, 如 `tex` 或 `enc files`。运行 `kpsewhich --help-formats` 以获取格式列表。
- `--mode=string`
将模式名称设置为 *string*; 这仅影响 ‘gf’ 和 ‘pk’ 的查找。没有缺省值: 任何模式都将被找到。
- `--must-exist`
尽一切可能找到文件, 特别包括在磁盘上搜索。默认情况下, 为了提高效率, 仅检查 `ls-R` 数据库。
- `--path=string`
沿路径 *string* 进行搜索 (通常以冒号分隔), 而不是从文件名中猜测搜索路径。支持 ‘//’ 和所有常见的扩展。选项 ‘--path’ 和 ‘--format’ 是互斥的。
- `--progrname=name`
将执行查找的程序名称设为 *name*。这会通过 `.progrname` 特性影响搜索路径。缺省值是 `kpsewhich`。
- `--show-path=name`
显示用于查找 *name* 类型文件的路径。和 ‘--format’ 选项一样, 可以使用扩展名 (`.pk`, `.vf`, 等等) 也可以使用全名。
- `--debug=num`
将调试选项 (等级) 设置为 *num*。

7.2.3 使用举例

现在我们看看实际使用 Kpathsea 的例子。这里是一个简单的搜索:

```
> kpsewhich article.cls
```

```
/usr/local/texmf-dist/tex/latex/base/article.cls
```

我们寻找的是 `article.cls` 文件。因为 `.cls` 后缀已经说明了文件的类型，所以我们不需要特别指明查找的是 `tex` 类型的文件（也就是要在 $\text{T}_{\text{E}}\text{X}$ 源文件目录下查找）。我们在 $\text{T}_{\text{E}}\text{X}$ Live 目录的 `'texmf-dist'` 下的 `tex/latex/base` 子目录找到了这个文件。与之类似，下列所有文件都顺利找到，因为其扩展名没有歧义。

```
> kpsewhich array.sty
/usr/local/texmf-dist/tex/latex/tools/array.sty
> kpsewhich latin1.def
/usr/local/texmf-dist/tex/latex/base/latin1.def
> kpsewhich size10.clo
/usr/local/texmf-dist/tex/latex/base/size10.clo
> kpsewhich small2e.tex
/usr/local/texmf-dist/tex/latex/base/small2e.tex
> kpsewhich tugboat.bib
/usr/local/texmf-dist/bibtex/bib/beebe/tugboat.bib
```

顺便提一下，最后一个是 $\text{BibT}_{\text{E}}\text{X}$ 的文献数据库，用于 *TUGboat* 文章。

```
> kpsewhich cmr10.pk
```

类型为 `.pk` 的字体位图字形文件被 `dvips` 和 `xdvi` 等显示程序使用。在这种情况下，不会返回任何内容，因为 $\text{T}_{\text{E}}\text{X}$ Live 中没有预生成的 Computer Modern `.pk` 文件——默认使用的是 Type 1 变体。

```
> kpsewhich wsuipa10.pk
/usr/local/texmf-var/fonts/pk/ljfour/public/wsuipa/wsuipa10.600pk
```

对于这个字体（来自华盛顿大学的一套音标字母表）就必须生成 `.pk` 文件了。我们默认的 METAFONT 模式是 `ljfour`，基础解析度是 600 dpi（每英寸点数），所以会得到这样一个文件。

```
> kpsewhich -dpi=300 wsuipa10.pk
```

在这种情况下，一旦指定我们感兴趣的分辨率是 300 dpi (`-dpi=300`) 时，就会发现系统里没有符合要求的文件。像 `dvips` 或 `xdvi` 这样的程序会使用 `mktexpk` 脚本去创建所需的 `.pk` 文件。

下面我们将注意力转向 `dvips` 的头文件和配置文件。首先看看最常用的一个，即用于 $\text{T}_{\text{E}}\text{X}$ 支持的一般序言文件 `tex.pro`，然后检查通用配置文件 (`config.ps`) 和 PostScript 字体映射文件 (`psfonts.map`)——从 2004 年开始，映射文件和编码文件都在 `texmf` 目录树下有其自己的搜索路径了。因为 `.ps` 后缀是模糊的，我们必须明确指定我们正在考虑的文件类型 (`dvips config`) 用于 `config.ps` 文件。

```
> kpsewhich tex.pro
/usr/local/texmf/dvips/base/tex.pro
> kpsewhich --format="dvips config" config.ps
/usr/local/texmf/dvips/config/config.ps
> kpsewhich psfonts.map
/usr/local/texmf/fonts/map/dvips/updmap/psfonts.map
```

现在我们仔细查看 URW Times PostScript 支持文件。在标准字体命名方案中，这些文件的前缀是 `'utm'`。我们首先查看的是配置文件，该文件包含映射文件的名称：

```
> kpsewhich --format="dvips config" config.utm
```

```
/usr/local/texmf-dist/dvips/psnfss/config.utm
```

这个文件的内容是

```
p +utm.map
```

即指向 `utm.map` 文件，也就是我们下一步要找的。

```
> kpsewhich utm.map
```

```
/usr/local/texmf-dist/fonts/map/dvips/times/utm.map
```

这个 `map` 文件定义了 URW 集中的 Type 1 PostScript 字体文件名。其内容如下所示 (我们只列出了其中一部分):

```
utmb8r NimbusRomNo9L-Medi    ... <utmb8a.pfb
utmbi8r NimbusRomNo9L-MediItal... <utmbi8a.pfb
utmr8r NimbusRomNo9L-Regu    ... <utmr8a.pfb
utmri8r NimbusRomNo9L-ReguItal... <utmri8a.pfb
utmb08r NimbusRomNo9L-Medi    ... <utmb8a.pfb
utmro8r NimbusRomNo9L-Regu    ... <utmr8a.pfb
```

例如，让我们以 Times Roman 实例 `utmr8a.pfb` 为例，搜索 Type 1 字体文件以找到它在 `texmf` 目录树中的位置：

```
> kpsewhich utmr8a.pfb
```

```
/usr/local/texmf-dist/fonts/type1/urw/times/utmr8a.pfb
```

从这些例子中可以清楚地看出，你可以轻松找到给定文件的位置。如果您怀疑某个文件的错误版本被意外使用，这一点尤其重要，因为 `kpsewhich` 将显示遇到的第一个文件。

7.2.4 调试操作

有时候，有必要调查一个程序如何解析文件引用。为了使这变得实际，`Kpathsea` 提供了各种级别的调试输出：

- 1 `stat` 调用 (磁盘上的查询)。在 `ls-R` 数据库及时更新的情况下几乎不会有什么输出。
- 2 对散列表的引用 (例如 `ls-R` 数据库，映射文件，配置文件)。
- 4 文件的打开与关闭操作。
- 8 `Kpathsea` 搜索的文件类型的通用路径信息。这有助于寻找针对某一文件的特定路径。
- 16 每个路径元素的目录列表 (只在本地磁盘上搜索时有用)。
- 32 文件搜索。
- 64 变量的值。

级别 `-1` 将启用上述所有选项，实际上这是最方便的设置方法了。

同样地，通过设置一组调试开关，可以使用 `dvips` 程序详细跟踪文件从哪里被读取。或者，当一个文件未找到时，调试跟踪会显示程序在查找该文件时搜索的目录，从而可以指示问题所在。

一般来说，由于大多数程序在内部调用 `Kpathsea` 库，可以通过使用 `KPATHSEA_DEBUG` 环境变量并将其设置为上述列表中描述的值 (或其组合) 来选择调试选项。

(Windows 用户请注意：因为在 Windows 下不容易把所有信息都重定向到固定的文件中，为了方便诊断，你可以临时设置 `SET KPATHSEA_DEBUG_OUTPUT=err.log`)。

```

debug:start search(file=texmf.cnf, must_exist=1, find_all=1,
  path=./usr/local/bin/texlive:/usr/local/bin:
    /usr/local/bin/texmf/web2c:/usr/local:
    /usr/local/texmf/web2c/././teTeX/TeX/texmf/web2c:).
kdebug:start search(file=ls-R, must_exist=1, find_all=1,
  path=~/.tex:/usr/local/texmf).
kdebug:search(ls-R) =>/usr/local/texmf/ls-R
kdebug:start search(file=aliases, must_exist=1, find_all=1,
  path=~/.tex:/usr/local/texmf).
kdebug:search(aliases) => /usr/local/texmf/aliases
kdebug:start search(file=config.ps, must_exist=0, find_all=0,
  path=./tex:!!/usr/local/texmf/dvips/).
kdebug:search(config.ps) => /usr/local/texmf/dvips/config/config.ps
kdebug:start search(file=/root/.dvipsrc, must_exist=0, find_all=0,
  path=./tex:!!/usr/local/texmf/dvips/).
search(file=/home/goossens/.dvipsrc, must_exist=1, find_all=0,
  path=./tex/dvips//:!!/usr/local/texmf/dvips/).
kdebug:search($HOME/.dvipsrc) =>
kdebug:start search(file=config.cms, must_exist=0, find_all=0,
  path=./tex/dvips//:!!/usr/local/texmf/dvips/).
kdebug:search(config.cms)
=>/usr/local/texmf/dvips/cms/config.cms

```

图 12: 寻找配置文件

让我们以一个简单的 L^AT_EX 源文件 `hello-world.tex` 为例，其内容如下：

```

\documentclass{article}
\begin{document}
Hello World!
\end{document}

```

这个小文件只使用了 `cmr10` 字体，我们可以看看 `dvips` 是如何生成 PostScript 文件的（我们希望使用 Type 1 版本的 Computer Modern 字体，所以使用了 `-Pcms` 参数）。

```
> dvips -d4100 hello-world -Pcms -o
```

此时将 `dvips` 的调试级别 4（表示字体路径）和 `Kpathsea` 的路径元素展开组合到一起（参见 `dvips` 参考手册）。（稍作整理的）输出见图 12。

`dvips` 首先定位其工作文件。首先找到 `texmf.cnf`，它定义了其他文件的搜索路径，然后是文件数据库 `ls-R`（用于优化文件搜索）和文件 `aliases`，它可以为同一个文件声明多个名称（例如，一个简短的类似 DOS 的 8.3 名称和一个更自然的较长版本）。接着，`dvips` 找到通用配置文件 `config.ps`，然后寻找自定义文件 `.dvipsrc`（在此情况下未找到）。最后，`dvips` 定位到计算机现代 PostScript 字体的配置文件 `config.cms`（这是通过 `dvips` 命令中的 `-Pcms` 选项启动的）。该文件包含定义字体的 T_EX、PostScript 和文件系统名称之间关系的映射文件列表。

```

> more /usr/local/texmf/dvips/cms/config.cms
p +ams.map
p +cms.map
p +cmbkm.map

```

```
p +amsbkm.map
```

因此, dvips 继续查找所有这些文件, 以及通用映射文件 `psfonts.map`, 该文件始终会被加载 (它包含常用 PostScript 字体的声明; 有关 PostScript 映射文件处理的更多详细信息, 请参见第 7.2.3 节的最后部分)。

这时候 dvips 向用户表示它的存在:

```
This is dvips(k) 5.92b Copyright 2002 Radical Eye Software
```

然后开始寻找 prolog 文件 `texc.pro`:

```
kdebug:start search(file=texc.pro, must_exist=0, find_all=0,
  path=.:~/tex/dvips/./:/usr/local/texmf/dvips/./:
    ~/tex/fonts/type1/./:/usr/local/texmf/fonts/type1/./).
kdebug:search(texc.pro) => /usr/local/texmf/dvips/base/texc.pro
```

找到相关文件后, dvips 输出日期和时间, 并告知我们它将生成文件 `hello-world.ps`, 然后它需要字体文件 `cmr10`, 并且后者被声明为“常驻 (resident) 的” (也就是不需要载入位图文件的字体):

```
TeX output 1998.02.26:1204' -> hello-world.ps
Defining font () cmr10 at 10.0pt
Font cmr10 <CMR10> is resident.
```

现在开始搜索文件 `cmr10.tfm`, 该文件已找到, 然后引用了更多的 prolog 文件 (此处未显示), 最终找到了 `cmr10.pfb` 这个 Type 1 字体, 并将它包含在输出文件中 (见最后一行)。

```
kdebug:start search(file=cmr10.tfm, must_exist=1, find_all=0,
  path=.:~/tex/fonts/tfm/./:/usr/local/texmf/fonts/tfm/./:
    /var/tex/fonts/tfm/./).
kdebug:search(cmr10.tfm) => /usr/local/texmf/fonts/tfm/public/cm/cmr10.tfm
kdebug:start search(file=texps.pro, must_exist=0, find_all=0,
  ...
<texps.pro>
kdebug:start search(file=cmr10.pfb, must_exist=0, find_all=0,
  path=.:~/tex/dvips/./:/usr/local/texmf/dvips/./:
    ~/tex/fonts/type1/./:/usr/local/texmf/fonts/type1/./).
kdebug:search(cmr10.pfb) => /usr/local/texmf/fonts/type1/public/cm/cmr10.pfb
<cmr10.pfb>[1]
```

7.3 运行时选项

Web2C 另一项有用的特性是可以通过 `Kpathsea` 读取的运行时文件 `texmf.cnf` 来控制一系列的内存参数 (尤其是数组的大小)。这些内存的设置可以在该文件的第三部分找到。比较重要的几个设置是:

main_memory $\text{T}_{\text{E}}\text{X}$ 、 METAFONT 和 MetaPost 可用的总内存 word 数。对于每个不同的设置, 你必须创建一个新的格式文件。例如, 您可以生成一个“巨大”版本的 $\text{T}_{\text{E}}\text{X}$, 并将格式文件命名为 `hugetex.fmt`。使用 `Kpathsea` 指定程序名称的标准方法时, 将从 `texmf.cnf` 中读取 `main_memory` 变量的特定值。

extra_mem_bot 为“大型” $\text{T}_{\text{E}}\text{X}$ 数据结构预留的额外空间: `boxes`, `glue`, `breakpoint` 等。尤其在使用 $\text{P}_{\text{I}}\text{C}_{\text{T}}\text{E}_{\text{X}}$ 时特别有用。

font_mem_size $\text{T}_{\text{E}}\text{X}$ 用于存储字体数据的 word 数。大致等于载入的所有 TFM 文件的总和。

`hash_extra` 为存储控制序列而设置的散列表的额外空间。默认的值是 600000。

此功能并不能真正替代动态数组和内存分配，但由于在当前的 $\text{T}_\text{E}\text{X}$ 源代码中实现这些功能极其困难，这些运行时参数提供了一种实用的折衷方案，允许一定的灵活性。

7.4 $\$TEXMFDOTDIR$

在上述许多地方，我们给出了以 `.` 开头的各种搜索路径（首先搜索当前目录），如下所示：

```
TEXINPUTS=.;$TEXMF/tex//
```

这是一种简化。我们在 $\text{T}_\text{E}\text{X}$ Live 分发的 `texmf.cnf` 文件使用 $\$TEXMFDOTDIR$ 而不仅是 `.`，比如：

```
TEXINPUTS=$TEXMFDOTDIR;$TEXMF/tex//
```

（在分发的文件中，第二个路径元素也比 $\$TEXMF/tex//$ 稍微复杂一点。但差别不大；而且我们这里讨论的是 $\$TEXMFDOTDIR$ 功能。）

在路径定义中使用变量 $\$TEXMFDOTDIR$ 而不是简单地使用 `.` 的原因纯粹是为了便于重载。例如，一份复杂的文档可能有分布在多个子目录中的很多源文件。为了解决这个问题，你可以将 $\$TEXMFDOTDIR$ 设置为 `./`（例如，在构建文档时在环境中设置），这样它们都会被搜索。（警告：不要默认使用 `./`；通常来说，在所有子目录中搜索任意文档的结果不理想，并且可能存在安全隐患。）

另一个例子是你可能完全不希望搜索当前路径，比如你安排所有的文件都通过具体的路径找到。那样你可以把 $\$TEXMFDOTDIR$ 设置为诸如 `/nonesuch` 这样不存在的目录。

$\$TEXMFDOTDIR$ 的默认值就是 `.`，正如在我们的 `texmf.cnf` 里设置的那样。

8 致谢

$\text{T}_\text{E}\text{X}$ Live 是在几乎所有 $\text{T}_\text{E}\text{X}$ 用户组织的协力下完成的。这个版本由 Karl Berry 监制。下面列出了过去和现在主要的贡献者：

- 英国、德国、荷兰和波兰的 $\text{T}_\text{E}\text{X}$ 用户组织（分别为 TUG, DANTE e.V., NTG, 和 GUST），他们为所在地区的 $\text{T}_\text{E}\text{X}$ 社群提供了必备的技术和管理基础设施。请加入本地的 $\text{T}_\text{E}\text{X}$ 用户组织！（参见 <https://tug.org/usergroups.html>。）
- CTAN 团队 (<https://ctan.org>)，负责分发 $\text{T}_\text{E}\text{X}$ Live 镜像并提供包更新的公共基础设施，这是 $\text{T}_\text{E}\text{X}$ Live 赖以运行的基础。
- Nelson Beebe，他为 $\text{T}_\text{E}\text{X}$ Live 提供了许多平台，自己也进行了详尽的测试，并在参考文献上有无与伦比的贡献。
- John Bowman，他对先进的图形程序 Asymptote 做了许多修改，使之能在 $\text{T}_\text{E}\text{X}$ Live 中工作。
- Peter Breitenlohner 和 ϵ - $\text{T}_\text{E}\text{X}$ 团队，他们创造了未来 $\text{T}_\text{E}\text{X}$ 的稳定基础。同时特别感谢 Peter，他为整个 $\text{T}_\text{E}\text{X}$ Live 中 GNU autotools 的使用提供了主要的帮助。Peter 在 2015 年 10 月去世，我们将持续工作以纪念他。
- Jin-Hwan Cho 和整个 DVIPDFM x 团队，他们创造了这个优秀的 DVI 输出程序，以及对配置问题及时的回应。
- Thomas Esser，如果没有他优秀的 $\text{te}_\text{T}_\text{E}\text{X}$ 套件， $\text{T}_\text{E}\text{X}$ Live 根本不可能存在。
- Michel Goossens，他一起编写了原始的文档。

- Eitan Gurari, 他不知疲倦地改进着的 $\text{T}_\text{E}\text{X}4\text{ht}$ 程序用于创建这份文档的 HTML 版本, 年复一年。Eitan 于 2009 年 6 月过早地离开了我们, 我们希望把这份文档献给它, 以志纪念。
- Hans Hagen, 他对 $\text{ConT}_\text{E}\text{Xt}$ 宏包 (<https://pragma-ade.com>) 做了许多测试和修改, 使之能够包含在 $\text{T}_\text{E}\text{X}$ Live 的框架下, 并持续推进 $\text{T}_\text{E}\text{X}$ 的开发。
- Hàn Thế Thành, Martin Schröder, 和 $\text{pdfT}_\text{E}\text{X}$ 团队 (<http://pdftex.org>), 他们持续不断地改进 $\text{T}_\text{E}\text{X}$ 的功能。
- Hartmut Henkel, 他对 $\text{pdfT}_\text{E}\text{X}$ $\text{LuaT}_\text{E}\text{X}$ 等程序的开发起到了重要的贡献。
- Shunshaku Hirata, 起始开发和坚持维护 DVIPDFM_x 。
- Taco Hoekwater, 他对 MetaPost 和 $(\text{Lua})\text{T}_\text{E}\text{X}$ (<http://luatex.org>) 重要的开发使之焕发新的活力。他还参与了将 $\text{ConT}_\text{E}\text{Xt}$ 融入 $\text{T}_\text{E}\text{X}$ Live、给 Kpathsea 添加多线程功能, 等等许多工作。
- Khaled Hosny, 他对 $\text{XeT}_\text{E}\text{X}$, DVIPDFM_x 和阿拉伯及其他字体的开发都有巨大的贡献。
- Paweł Jackowski, 他创建了 Windows 下的安装程序 tlpm , 和 Tomasz Łuczak, 因为他的 tlpmgui 被用在了以前的版本。
- Akira Kakuto, 感谢他通过 W32TEX 和 W64TEX 发行版为日文 $\text{T}_\text{E}\text{X}$ (<http://w32tex.org>) 提供 Windows 二进制文件, 以及其他许多开发贡献。
- Jonathan Kew, 他开发了非凡的 $\text{XeT}_\text{E}\text{X}$ 引擎并花了大量时间和精力集成到 $\text{T}_\text{E}\text{X}$ Live 中。以及 $\text{MacT}_\text{E}\text{X}$ 安装程序的早期版本, 还有我们推荐的前端 $\text{T}_\text{E}\text{Xworks}$ 。
- Hironori Kitagawa, 对 $\text{pT}_\text{E}\text{X}$ 和相关的支持进行了大量的工作。
- Dick Koch, 他维护了 $\text{MacT}_\text{E}\text{X}$ (<https://tug.org/mactex>) 这个和 $\text{T}_\text{E}\text{X}$ Live 联系非常紧密的发行版本, 并在此过程中保持极好的乐观态度。
- Reinhard Kotucha, 他对 $\text{T}_\text{E}\text{X}$ Live 2008 的基础架构和安装程序起到了重要贡献, 以及 Windows 下的研究性工作, getnonfreefonts 脚本, 等等。
- Siep Kroonenberg, 也因为他对 $\text{T}_\text{E}\text{X}$ Live 2008 基础架构和安装程序的重要贡献, 尤其是在 Windows 上。他还花了大量的时间更新手册, 介绍了这些特性。
- Clerk Ma, 对引擎的 bug 修正和扩展。
- Mojca Miklavc, 为她在 $\text{ConT}_\text{E}\text{Xt}$ 上的工作、编译大量二进制文件和很多其他的工作。
- Heiko Oberdiek, 因为他的 epstopdf 包和许多其他的工作, 压缩巨大的 pst-geo 数据文件使我们得以包含它们, 最重要的还是因为他在 hyperref 宏包上优秀的工作。
- Phelype Oleinik, 对 2020 年在多个引擎上实现组分隔的 $\backslash\text{input}$ 命令, 以及更多的工作。
- Petr Olšák, 他非常认真地协调和检查所有的捷克语和斯洛伐克语资料。
- Toshio Oshima, 他提供了 Windows 下的 dviout 预览工具。
- Manuel Pégourié-Gonnard, 他对宏包更新、文档改进以及 texdoc 开发上的努力。
- Fabrice Popineau, 感谢其在 $\text{T}_\text{E}\text{X}$ Live 中提供的原始 Windows 支持以及对法文文档的工作。
- Norbert Preining, 他是当前 $\text{T}_\text{E}\text{X}$ Live 基础架构和安装程序的总设计师, 还负责协调了 Debian 版本的 $\text{T}_\text{E}\text{X}$ Live 工作 (和 Frank Küster 一起), 还有很多其他的工作。
- Sebastian Rahtz, 他是 $\text{T}_\text{E}\text{X}$ Live 的创始者, 并维护了它许多年。Sebastian 在 2016 年 3 月去世, 我们用持续的工作来纪念他。
- Luigi Scarso, 他坚持开发 MetaPost , $\text{LuaT}_\text{E}\text{X}$ 等等许多。
- Andreas Scherer, 感谢其为 cwebbin ($\text{T}_\text{E}\text{X}$ Live 中使用的 CWEB 实现) 所做的工作, 以及对原始 CWEB 的持续维护。
- Tim Stewart, 感谢其制作了 torrents 种子文件。
- Takuji Tanaka, 感谢他对 $(\text{e})\text{u}\text{pT}_\text{E}\text{X}$ 的维护及相关支持。
- Tomasz Trzeciak, 他为 Windows 开发提供了广泛的帮助。
- Vladimir Volovich, 他很大地帮助解决了许多移植和维护的问题, 尤其是令我们能够将 xindy 包含进来。
- Staszek Wawrykiewicz, $\text{T}_\text{E}\text{X}$ Live 的一个主要的测试人员, 同时还是许多重要波兰语支持的协调人员: 字体、Windows 安装, 和其他许多工作。Staszek 在 2018 年 2 月去世了, 我们将继续工作

以纪念他。

- Olaf Weber, 他在以前几年对 Web2C 耐心的维护。
- Gerben Wierda, 他创建和维护了原来的 macOS 支持。
- Graham Williams, 他是 T_EX Catalogue 的发起者。
- Joseph Wright, 他为了让多个引擎获得相同的 primitive 功能进行了大量的工作。
- Hironobu Yamashita, 对 pT_EX 和相关的支持进行了大量的工作。

二进制版本的编译器: Marc Baudoin (amd64-netbsd, i386-netbsd), Ken Brown (x86_64-cygwin), Akira Kakuto (windows), Dick Koch (universal-darwin), Mojca Miklavc (x86_64-darwinlegacy), Norbert Preining (aarch64-linux, armhf-linux, i386-freebsd, amd64-freebsd, i386-linux, x86_64-linux, i386-solaris, x86_64-solaris, x86_64-linuxmusl). 关于 T_EX Live 编译进程的信息, 请查阅 <https://tug.org/texlive/build.html>.

这份手册的翻译者: Takuto Asakura (Japanese), Denis Bitouzé & Patrick Bideault (法语), Carlos Enriquez Figueras (西班牙语), Jiang Jiang, Jinsong Zhao, Yue Wang, Helin Gai & Ran Wang (中文), Nikola Lečić (塞尔维亚语), Marco Pallante & Carla Maggi (意大利语), Petr Sojka & Jan Busa (捷克/斯洛伐克语), Boris Veytsman & Vladimir Lomov (俄语), Uwe Ziegenhagen (德语)。T_EX Live 文档的主页是 <https://tug.org/texlive/doc.html>。

当然, 最重要的感谢应该给予 Donald Knuth, 感谢他发明了 T_EX, 也感谢他将 T_EX 赠与全世界。

9 发行历史

9.1 过去

1993 年末荷兰 T_EX 用户组开始为 MS-DOS 用户开发 4AllT_EX CD 时, 我们就开始了相关的讨论, 并希望在此时为所有的操作系统提供一个单一的、合理的 CD。当时那是一个过于宏伟的目标, 但的确滋生了非常成功的 4AllT_EX CD, 同时 TUG 技术委员会工作组也开始设计 T_EX 目录结构 (<https://tug.org/tds>), 以指明如何创建一套一致而可控的集合, 囊括所有 T_EX 相关的文件。TDS 的完整草案在 1995 年 12 月的 TUGboat 上出版, 并初步确定期望的产品将是在 CD 上出现的范例结构。你现在使用的这个发行版正是工作组审议的直接结果。4AllT_EX CD 的成功也说明如果有一个类似这样的易于使用的系统, 对 Unix 用户肯定很有帮助, 这是 T_EX Live 最主要的出发点。

我们在 1995 年秋天开始尝试构建一个新的 CD (基于 TDS), 并很快发现 Thomas Esser 的 teT_EX 已经是比较理想的配置, 并因为在它构建时就已经考虑了跨文件系统的兼容性问题, 也已具有多平台支持。Thomas 同意帮助我们, 并在 1996 年初开始了正式的工作。第一版是在 1996 年五月发行的。到 1997 年初, Karl Berry 完成了 Web2C 的一个重大的更新版本, 将几乎所有 Thomas Esser 加入 teT_EX 的特性囊括在内, 这样我们决定在 teT_EX 的 texconfig 脚本的辅助下, 基于标准的 Web2C 来制作第二版的 CD。第 3 版的 CD 基于 Olaf Weber 完成的 Web2C 的一个重大修正版本, 7.2。与此同时, teT_EX 的一个新版本出现了, T_EX Live 也包含了其中绝大多数特性。第 4 版依照上面的模式进行, 使用了新版本的 teT_EX 和新版本的 Web2C (7.3)。感谢 Fabrice Popineau 的帮助, 系统此时也包括了完整的 Windows 下的配置。

在第 5 版 (2000 年 3 月) 中检查并修正了 CD 的许多部分, 更新了数百个软件包。软件包的详细说明现在存放在 XML 文件中。不过 T_EX Live 5 的首要变化还是移除了所有的非自由软件。T_EX Live 的所有部分现在都在向 Debian Free Software Guidelines (<https://debian.org/intro/free>) 兼容的方向改进, 我们尽最大努力检查了所有软件包的授权协议, 欢迎为我们指出错误。

第 6 版 (2001 年 7 月) 更新了许多内容。最重大的一项是新的安装形式, 用户可以更精确地选择所需的软件集合。与语言相关的集合也重新组织过了, 这样一来, 选定某个语言集合时会自动安装宏包、字体等文件, 并自动设置好 `language.dat`。

2002 年出现的第 7 版里显著的更新是添加了 macOS 的支持, 还有大量各类宏包和程序的更新。这个版本的一个重要的目标是将源代码重新与 `teTeX` 集成, 因为在第 5 和第 6 版中它们偏离得太远了。

9.1.1 2003

2003 年, 在更新和增添持续不断到来的情况下, 我们发现 `TeX Live` 已经过于庞大, 无法在一张 CD 中容纳, 于是将其切分为三套不同的发行版 (参见第 2.1 节, p.6)。此外:

- 在 `LATEX` 团队的要求下, 我们将 `latex` 和 `pdflatex` 命令改为使用 `ε-TeX` 引擎 (参见 p.8)。
- 包含了新的 Latin Modern 字体 (并推荐使用)。
- 因为不再有人拥有 (或主动提供) 用于编译新的二进制程序的硬件, 去除了 Alpha OSF 的支持 (先前已经去除了 HP-UX 的支持)。
- Windows 下的安装有很大改变, 首次提供了基于 XEmacs 的集成环境。
- Windows 下重要的辅助性程序 (Perl, Ghostscript, ImageMagick, Ispell) 现在放在 `TeX Live` 的安装目录。
- `dvips`, `dvipdfm` 和 `pdftex` 使用的字体映射文件现在通过 `updmap` 这套新程序生成, 并安装到 `texmf/fonts/map` 目录下。
- `TeX`, `METAFONT`, 和 `MetaPost` 现在缺省直接输出大多数的输入字符 (位置 32 及其以上) (比如通过 `\write`), 包括输出到文件、日志和终端上。也就是说, 不再使用 `^^` 标记来转换。在 `TeX Live 7` 中是否转换根据系统区域 (locale) 设置而定, 而这一版里 locale 设置不再影响 `TeX` 程序的行为, 所以如果你需要 `^^` 形式的输出, 请将 `texmf/web2c/cp8bit.tcx` 文件改名。(后续版本将提供更简洁的方式来控制。)
- 对文档作了大量更新。
- 最后, 因为版本号增长得太快, 现在简单地使用年份来标识版本: `TeX Live 2003`。

9.1.2 2004

2004 年有许多改变:

- 如果你在本地安装的字体时涉及了 `.map` 或 `.enc` (附带这种文件的可能性很小) 辅助文件, 可能需要转移这些文件的位置。
现在根据 `TEXFONTMAPS` 变量中的路径设置, 只在 (所有 `texmf` 目录树下的) `fonts/map` 子目录下搜索 `.map` 文件。与之类似, `.enc` 文件现在只在 `fonts/enc` 目录下搜索, 根据 `ENCFONTs` 变量中的路径设置。如果遇到有问题的文件, `updmap` 会提出警告。
关于这种搜索方式的其他信息, 请参见 <https://tug.org/texlive/mapenc.html>。
- 因为有人可能更愿意使用 `MiKTeX` 而非 `Web2C` 系统, `TeX Collection` 现在包含了一套基于 `MiKTeX` 的可安装 CD, 参见第 2 节 (p.6)。
- 在原来旧版本 `TeX Live` 中单一的 `texmf` 目录树被分拆为三个: `texmf`, `texmf-dist`, 和 `texmf-doc`。参见第 2.2 节 (p.6) 及各目录下的 `README` 文件。
- 所有 `TeX` 输入文件现在统一收集到了 `texmf*` 下的 `tex` 子目录中, 不再分散在各个 `tex`, `etex`, `pdftex`, `pdfetex` 目录。见 `texmf-dist/doc/english/tds/tds.html#Extensions`。
- 辅助性脚本 (并非直接提供给用户调用的) 现在放在 `texmf*` 目录树下新的 `scripts` 子目录中, 并可以通过 `kpsewhich -format=texmfscripts` 来搜索。如果你的程序调用了这些脚本, 必须修改

路径。参见 texmf-dist/doc/english/tds/tds.html#Scripts。

- 几乎所有格式，都用 `cp227.tcx` 这个转换文件将大多数的可见 (printable) 字符保留下来，而不再使用 `^^` 标识来转换这些字符。具体而言，在位置 32–256 的字符，加上 `tab`, `vertical tab`, 和 `form feed` 字符都作为可见字符而不再转换。例外情况是 `plain TEX` (只将位置 32–126 的字符视为可见), `ConTEXt` (0–255 都视为可见) 和与 Ω 相关的格式。缺省的情况几乎与 `TEX Live 2003` 完全一致，但通过更简洁的方式实现，并允许更多定制。参见 texmf-dist/doc/web2c/web2c.html#TCX-files。(另外，如果遇到 Unicode 输入，`TEX` 可能会在显示错误上下文时输出半个字符，因为它是基于字节流来处理输入的。)
- `pdftex` 现在是除 (plain) `tex` 外所有格式的默认引擎 (当然以 `latex` 这种方式运行时它还是生成 DVI)。这样一来，至少 `pdftex` 的微调排版 (microtypographic) 技术可以在 `LATEX`, `ConTEXt` 等格式中使用，另外 ε -`TEX` 的特性也包含在其中 (texmf-dist/doc/etex/base/)。这还说明比以前任何时候都更有必要使用 (对 `plain` 和 `LATEX` 都适用的) `ifpdf` 宏包或其类似代码，因为只检查 `\pdfoutput` 或其他原语是否已经定义不再是判断是否处于 PDF 输出状态的可靠方法。我们在这一年尽可能地保持向下兼容，但以后即使在输出 DVI 时 `\pdfoutput` 也可能已经定义。
- `pdfTEX` (<http://pdftex.org>) 新增了许多特性：
 - 可以使用 `\pdfmapfile` 和 `\pdfmapline` 来在单独文档内指定字体映射文件。
 - 可以更方便地使用排版微调 (Microtypographic) 和字体延展 (font expansion) 技术了。
<http://www.ntg.nl/pipermail/ntg-pdftex/2004-May/000504.html>
 - 原来使用专有格式的配置文件 `pdftex.cfg` 里的选项现在都必须改用 `TEX` 原语来设置，通常放在 `pdftexconfig.tex` 里面，不再支持 `pdftex.cfg` 的配置方式。每次修改 `pdftexconfig.tex` 之后都必须重新生成 `.fmt` 文件。
 - 参见 `pdfTEX` 手册以了解更多信息: texmf-dist/doc/pdftex/manual/pdftex-a.pdf。
- `tex` (以及 `mf` 和 `mpost`) 中的 `\input` 原语现在支持通过双引号来引用包含空格和特殊字符的文件。一个典型的例子如下：


```
\input "filename with spaces"    % plain
\input{"filename with spaces"}    % latex
```

 参阅 Web2C 文档以了解更多信息: texmf-dist/doc/web2c/。
- `encTEX` 的支持现在已被包含在 Web2C 中，因而所有 `TEX` 程序都可以通过 `-enc` 参数启用这一支持——前提是构建好了格式文件。`encTEX` 提供了对输入输出通用的重新编码功能，实现对 Unicode (以 UTF-8 编码的形式) 的完整支持。参见 texmf-dist/doc/generic/encTeX/ 和 <http://olsak.net/encTeX.html>。
- 提供了 Aleph 这套新的 `TEX` 引擎，它将 ε -`TEX` 和 Ω 合并到了一起。关于 Aleph 的部分信息可以在 texmf-dist/doc/aleph/base 和 <https://texfaq.org/FAQ-enginedev> 找到。Aleph 的 `LATEX` 格式文件称做 `lamed`。
- 最新发布的 `LATEX` 包含了是新版的 LPPL 授权协议——这一协议已被 Debian 首肯。`LATEX` 其他的更新请见 texmf-dist/doc/latex/base 下的 `ltnews` 文件。
- 包含了一个叫做 `dvipng` 的新程序，用于将 DVI 转换为 PNG 图像文件。参见 <https://ctan.org/pkg/dvipng>。
- 我们在作者 (Claudio Beccari) 的同意下，将 `cbgreek` 包含的字体数量减少到中等。去除了不可见、轮廓和透明版本的字体，这些字体几乎很少用到。而我们的光盘镜像需要空间。完整版本当然还是在 CTAN 提供 (<https://ctan.org/pkg/cbgreek-complete>)。
- 去掉了 `oxdvi`，改为只使用 `xdvi`。
- 不再为 `tex`, `mf`, 和 `mpost` 程序创建 `ini` 和 `vir` 开头的命令链接，比如 `initex`。`ini` 的功能早在几年前就通过 `-ini` 命令行参数提供了。
- 去掉了 `i386-openbsd` 平台的支持。因为在 BSD Ports 系统中已经包含了 `tetex` 软件包，而

GNU/Linux 和 FreeBSD 下的二进制版本都已存在，所以志愿者的时间可以花在别的地方了。

- 至少在 sparc-solaris 平台下，你必须设置好 LD_LIBRARY_PATH 环境变量才能执行 t1utils 包含的程序。因为这些程序是使用 C++ 编写的，其运行时库没有固定的位置。(这一情况并非在 2004 版中首次出现，但现在才写入文档) 与之类似，mips-irix 平台下需要用到 MIPSpro 7.4 运行时库。

9.1.3 2005

2005 年一如往常，宏包和程序都有大量的更新。底层结构和 2004 年相比保持了稳定，不过仍然存在一些变化。

- 引入了新的 texconfig-sys, updmap-sys, 和 fmtutil-sys 安装脚本，用于修改系统目录下的配置。而原有的 texconfig, updmap, 和 fmtutil 则用于修改针对单个用户的文件 (放在 \$HOME/.texlive2005 目录下的)。
- 增加了对应的 TEXMFCONFIG 和 TEXMFSYSCONFIG 变量，分别用于设置针对用户和系统的，专门存放配置文件的目录树。所以你需要将个人使用的 fmtutil.cnf 和 updmap.cfg 放到合适位置。不过还有一种方法是在 texmf.cnf 里边重新定义 TEXMFCONFIG 或 TEXMFSYSCONFIG 变量。无论如何，这两个值对应的实际目录都必须正确存在。参见第 7 页的第 2.3 节。
- 虽然我们在上一年就已经使用了 pdfetex 作为输出程序，但在它输出 DVI 格式时会禁用 \pdfoutput 等原语 (primitive)。这一年，我们按照预期计划取消了这一兼容性限制。所以，如果你的文档里使用了 \ifx\pdfoutput\undefined 这样的语句来判断是否正在 PDF 输出模式下，现在就必须修改了。你可以使用 ifpdf.sty 宏包 (对 plain TeX 和 L^AT_EX 都适用) 来判断，或者仿照这个文件里的判断原理自己写一个。
- 上一年，我们将格式文件的输出改成了和这些文件本身一样的 8 位字符。在你需要的情况下，可以使用新的 TCX 文件 empty.tcx 来获得原有的 ^^ 表示方式。例如：

```
latex --translate-file=empty.tcx yourfile.tex
```

- 新增了用于转换 DVI 为 PDF 的 dvipdfmx 程序，这是 dvipdfm 的一个比较活跃更新的版本 (我们仍然提供 dvipdfm，但不建议你继续使用)。
- 新增了叫 pdfopen 和 pdfclose 的两个程序，用于控制 Adobe Acrobat/Reader 在不重启程序的情况下重新载入 PDF 文件。(其他的 PDF 阅读器，如 xpdf, gv, 和 gsvie, 都不会遇到这个问题。)
- 为了保持一致性，将 HOMETEXMF 和 VARTEXMF 环境变量分别更名为 TEXMFHOME 和 TEXMFSYSVAR。还有一个针对单独用户的 TEXMFVAR 环境变量可用。参见上面的第一点。

9.1.4 2006–2007

2006–2007 年，T_EX Live 的一个重大变化是增加了 XeT_EX，以 xetex 和 xelatex 程序的形式提供。请参见 <https://scripts.sil.org/xetex>。

MetaPost 也有可观的更新，并计划在未来实现更多的改进 (<https://tug.org/metapost/articles>)，pdfT_EX 同样如此 (<https://tug.org/applications/pdftex>)。

T_EX .fmt (缓存格式) 文件和用于 MetaPost 和 METAFONT 的类似文件现在存储在 texmf/web2c 的子目录中而不直接放在 texmf/web2c 目录下 (不过考虑到现有的 .fmt 文件，直接放置在这个目录下的文件仍然能被搜索到)。子目录的名称是根据当前使用的“引擎”决定的，比如 tex 或 pdftex 或 xetex。这个变化不会对日常使用带来任何影响。

(plain) `tex` 程序不再通过读取 `%&` 开头的第一行来决定执行何种格式，而遵循纯粹的 Knuth 风格 \TeX 的传统。(L \TeX 和其他所有的程序仍然读取 `%&` 开头的行。)

当然，和往常一样，这一年里你能看到成百上千的宏包与程序得到更新。也和往常一样，进一步的更新请使用 CTAN (<https://ctan.org>)。

从内部角度上看，源代码树现在改为使用 Subversion 管理，并在我们的主页上提供了到 Web 界面的链接，用于浏览代码树。我们希望它能成为未来几年中稳定的开发平台。

末了，2006 年五月 Thomas Esser 宣布他将停止 `te\TeX` (<https://tug.org/tetex>) 的更新。这样一来，大家对 \TeX Live 的兴趣大增，尤其是在 GNU/Linux 发行版中。(\TeX Live 提供了一套新的 `tetex` 安装方案，几乎和原有的 `te\TeX` 毫无二致。) 我们希望这些变化将最终转换为对整个 \TeX 环境的改进，从而每个人都会受益。

9.1.5 2008

在 2008 年，整个 \TeX Live 的基础架构都重新设计过并重新实现了。安装的完整信息被存放在 `tlpkg/texlive.tlpdb` 这个纯文本文件中。

除了许多其他的功能外，我们终于可以通过 Internet 在安装后更新 \TeX Live 了，这是 MiK \TeX 早已提供了许多年的特性。我们希望能定期更新 CTAN 上新发布的软件包。

包含了一个重要的新引擎 Lua \TeX (<http://luatex.org>)，除了在排版上灵活性更上一层楼以外，它还提供了一个优秀的脚本语言供 \TeX 文档内外使用。

对 Windows 和基于 Unix 系统的支持现在要一致得多了。尤其是大部分 Perl 和 Lua 脚本都能在 Windows 下使用，通过 \TeX Live 内部包含的 Perl。

新的 `tlmgr` 脚本 (第 5 节) 是在初始安装后管理 \TeX Live 的通用界面。它处理了软件包更新和后续的格式文件、`map` 文件和语言文件的重新生成，并可选地包含了本地添加的内容。

随着 `tlmgr` 到来，禁用了 `texconfig` 中编辑格式和断字配置文件的功能。

今年还提供了大多数平台的 `xindy` 索引生成程序 (<http://xindy.sourceforge.net>)。

`kpsewhich` 现在可以报告给定文件的所有匹配 (`--all` 参数) 并限制列出特定目录下的匹配 (`--subdir` 参数)。

`dvipdfmx` 程序现在支持用 `extractbb` 命令来解析 bounding box 信息，这是 `dvipdfm` 包含但未曾出现在 `dvipdfmx` 中的最后一个功能。

去除了 Times-Roman, Helvetica 等字体别名。不同的宏包对它们的理解不同 (尤其是编码的处理不一样)，所以没有什么好的办法能解决。

去除了 `platex` 格式文件，以避免与日文 `platex` 的命名冲突，`polski` 宏包现在被用作主要的波兰语支持。

从内部而言 WEB 字符串 `pool` 文件被编译进了二进制文件中，这样可以方便升级。

最终 Donald Knuth 在他的 ‘ \TeX tuneup of 2008’ 中的更新也被包含在这次发布中。参见 <https://tug.org/TUGboat/Articles/tb29-2/tb92knut.pdf>。

9.1.6 2009

在 2009 年, 为了充分利用 Lua \TeX 的 OpenType 支持等特性, Lua(L^A) \TeX 的默认输出格式是 PDF。新增叫做 `dviluatex` 和 `dvilualatex` 的这两个命令会以 DVI 输出方式运行 Lua \TeX 。Lua \TeX 的主页在 <http://luatex.org>。

在与 Omega 的作者讨论后, 原来的 Omega 引擎和 Lambda 格式文件被去掉了。更新后的 Alpeh 和 Lamed 仍然在, 同时保留的还有 Omega 实用工具。

包含了新版本的 AMS Type 1 字体, 包括 Computer Modern: 其中部分字形随 Knuth 多年以来修改的 MetaFont 源代码更新, hinting 信息也更新了。Euler 字体也整个由 Hermann Zapf 重新绘制了一遍 (参见 <https://tug.org/TUGboat/Articles/tb29-2/tb92hagen-euler.pdf>)。不过上述变化并没有改变字体的 metrics 文件。AMS 字体的主页在 <https://www.ams.org/tex/amsfonts.html>。

现在 Windows 和 Mac \TeX 都包含了新的 GUI 前端 \TeX works。至于其他的平台和更多的信息, 请参见 \TeX works 的主页, <https://tug.org/texworks>。设计这个跨平台前端的灵感来自于 macOS 下的 TeXShop 编辑器, 目标就是易用。

在许多平台下包含了 Asymptote 图形程序, 它实现了一套与 MetaPost 约略相似的文本图形描述语言, 但包含了先进的 3D 支持等其他特性。它的主页在 <https://asymptote.sourceforge.net>。

单独的 `dvipdfm` 程序已被 `dvipdfmx` 所替代, 如果以 `dvipdfm` 这个名字调用的时候, 后者会以一种特殊的兼容性模式运行。`dvipdfmx` 包含了 CJK 支持, 并包含了多年以来在 `dvipdfm` 基础上的许多修正。

现在包括了 `cygwin` 和 `i386-netbsd` 平台下的可执行文件, 而我们建议 OpenBSD 和 FreeBSD 的用户使用他们自己的包管理系统提供的 \TeX , 另外这也是因为要编译出能在多种版本下都工作的二进制程序有些困难。

一些更不起眼的更新: 我们现在使用 `xz` 这套稳定的压缩方式来替代原有的 `lzma` (<https://tukaani.org/xz/>); 在不和现有变量名冲突的情况下允许文件中使用 `$` 字符; `Kpathsea` 库现在支持多线程了 (其中用到了 `MetaPost`); 整个 \TeX Live 的编译现在基于 `Automake` 了。

对过去历史的最终一点提示: 所有版本的 \TeX Live, 包括 CD 标签这些附属材料, 都在 <ftp://tug.org/historic/systems/texlive> 提供。

9.1.7 2010

在 2010 年, 缺省的 PDF 输出版本现在是 1.5, 以支持更多压缩。这对所有输出 PDF 的 \TeX 引擎生效, 也对 `dvipdfmx` 有效。载入 `pdf14` L^A \TeX 宏包将改回 PDF 1.4, 或者设置 `\pdfminorversion=4`。

在载入了 L^A \TeX `graphics.cfg` 配置文件, 而且输出的是 PDF 时, `pdf(LA) \TeX` 现在自动地将请求的封装 PostScript (EPS) 文件转换为 PDF, 通过 `epstopdf` 宏包。默认选项是为了避免覆盖任何手工创建的 PDF 文件, 但你也可以将 `\newcommand{\DoNotLoadEpstopdf}{} (或 \def...` 放在 `\documentclass` 声明前以避免载入 `epstopdf`。如果使用了 `pst-pdf` 宏包, 也不会载入它。要了解更多的细节, 参见 `epstopdf` 宏包的文档 (<https://ctan.org/pkg/epstopdf-pkg>)。

一个相关的变化是通过 `\write18` 特性从 \TeX 执行少量外部命令, 现在默认启用了。这些命令是 `repstopdf`, `makeindex`, `kpsewhich`, `bibtex`, 和 `bibtex8`; 这个列表定义在 `texmf.cnf`。必须禁用这些外部命令的环境可以通过安装程序取消这个选项 (参见第 3.2.4 节), 或者在安装后通过运行 `tlmgr conf texmf shell_escape 0` 覆盖这个值。

另一个相关的变化是 BibTeX 和 Makeindex 现在默认会拒绝往任意目录写入输出文件了 (类似 TeX 本身)。这样使得它们能被启用给受限的 \write18 使用。要修改这个行为, 可以设置 TEXMFOUTPUT 环境变量, 或者修改 openout_any 设置。

XeTeX 现在支持与 pdfTeX 一样的 margin kerning。 (Font expansion 现在还不支持。)

默认情况下, tlmgr 现在回给每个更新的包保存一个备份 (tlmgr option autobackup 1), 所以损坏的包更新可以简单地通过 tlmgr restore 恢复。如果你在安装后要执行更新, 但没有足够的磁盘空间来保存备份, 可以运行 tlmgr option autobackup 0。

包含了这些新的程序: 用于排版日文的 pTeX 引擎和相关实用工具; 支持了 Unicode BibTeX 的 BibTeXU 程序; chktex 实用工具 (<https://baruch.ev-en.org/proj/chktex>) 用来检查 (La)TeX 文档; dvisvgm (<https://dvisvgm.de>) 是 DVI 到 SVG 格式转换器。

现在包含了这些新平台的可执行文件: amd64-freebsd, amd64-kfreebsd, i386-freebsd, i386-kfreebsd, x86_64-darwin, x86_64-solaris。

我们忘记注明的一个 TeX Live 2009 的修改是: 许多 TeX4ht 相关的可执行文件 (<https://tug.org/tex4ht>) 被从二进制目录删除了。通用的 mk4ht 程序可以用来运行这些 tex4ht 组合的任意一种。

最后, TeX Collection DVD 上的 TeX Live 发行不能再直接执行了 (听起来很奇怪)。单张 DVD 已经没有足够的空间了。一个优点是物理 DVD 的安装将会快很多。

9.1.8 2011

macOS 二进制程序 (universal-darwin 和 x86_64-darwin) 现在只能在 Leopard 及以后版本上工作; Panther 和 Tiger 都不再支持了。

用于参考文献处理的 biber 程序在常见平台下都已包含。它的分发和 biblatex 宏包紧密相关的, 这个宏包完全重新实现了 LaTeX 提供的参考文献机制。

MetaPost (mpost) 程序不再创建或者使用 .mem 文件了。所需的文件, 比如 plain.mp, 会在每次运行时读入。这个变化和将 MetaPost 作为一个库有关, 这是另一个重要而用户不会注意的变化。

updmap 的 Perl 实现, 先前只在 Windows 下使用, 现在被更新并用于所有平台了。这不应该造成任何用户可见的变化, 除了它变得快得多。

恢复了 initex 和 inifm 程序。(但不包含其他的 ini* 变种。)

9.1.9 2012

tlmgr 支持了从多个网络仓库更新。tlmgr help 输出中的多仓库一节有更多信息。

对于 xetex 和 xelatex, \XeTeXdashbreakstate 参数被缺省设置为 1。这允许了 em-dash 和 en-dash 后的换行, 这和 plain TeX, LaTeX, LuaTeX 现在的行为一致。现有的 XeTeX 文档如果需要保持完美的换行兼容性则需要显式设置 \XeTeXdashbreakstate 为 0。

pdftex 和 dvips 等生成的输出文件现在可以超过 2 GB 了。

35 标准 PostScripts 字体现在在 dvips 输出中缺省保函了, 因为现在有太多这些字体的不同版本了。

默认设置的 \write18 受限执行模式中, mpost 成为了一个允许的程序。

`../texmf-local` 下现在也会出现一个 `texmf.cnf` 文件, 比如 `/usr/local/texlive/texmf-local/web2c/texmf.cnf`。

`updmap` 脚本现在读取的是针对各个目录树的 `updmap.cfg`, 而不再是全局的配置文件。这个修改应该不容易发现, 除非你直接编辑过 `updmap.cfg` 文件。`updmap --help` 输出中有更多信息。

平台: 增加了 `armel-linux` 和 `mipsel-linux` 平台; `sparc-linux` 和 `i386-netbsd` 不再出现在主发行中。

9.1.10 2013

发行版布局: 为了简化, 顶层的 `texmf/` 目录被并入 `texmf-dist/`。现在 `TEXMFMAIN` 和 `TEXMFDIST` 这两个 `Kpathsea` 变量都被指向 `texmf-dist` 了。

为了简化安装合并了许多小的语言集合。

MetaPost: 加入对 PNG 输出和 (IEEE 双精度) 浮点数的原生支持。

Lua_{TeX}: 升级到 Lua 5.2, 包含一个新的库 (`pdfscanner`) 来处理外部 PDF 页面内容, 以及其他功能 (见主页)。

Xe_{TeX} (见其主页了解更多信息):

- 使用 HarfBuzz 库替代 ICU 进行字体排版。(仍然使用 ICU 来支持输入编码、双向排版, 以及可选的 Unicode 断行。)
- 使用 Graphite2 和 HarfBuzz 来替代 SilGraphite 进行 Graphite 排版。
- 在 Mac 上, 使用 Core Text 替代 (Apple 不再建议使用的) ATSUI。
- 在名称相同的情况下优先使用 TrueType/OpenType 字体而不是 Type1 字体。
- 修正偶尔出现的 Xe_{TeX} 和 `xdvipdfmx` 字体查找不匹配的问题。
- 支持 OpenType math 间距调整。

xdvi: 现在使用 FreeType 替代 `t1lib` 进行字体渲染。

`microtype.sty`: 对 Xe_{TeX} 的部分支持 (`protrusion`) 和对 Lua_{TeX} 的支持 (`protrusion`, `font expansion`, `tracking`), 已经其他改进。

`tlmgr`: 新的 `pinning` 操作以方便配置多个仓库; 参见 `tlmgr --help` 的对应章节, 或者在线的 <https://tug.org/texlive/doc/tlmgr.html#MULTIPLE-REPOSITORIES>。

平台: `armhf-linux`, `mips-irix`, `i386-netbsd`, 和 `amd64-netbsd` 被重新加入; `powerpc-aix` 被去除。

9.1.11 2014

2014 年我们收到了 Knuth 的又一个 _{TeX} 修正, 这影响了所有的引擎, 但可能唯一可见的变化是在启动时恢复显示的 `preloaded format` 字符串。根据 Knuth 的说法, 这个字符串现在反应的时在启动时应该被默认载入的格式, 而不是在二进制程序中预载入的未 `dump` 格式, 该格式可能被很多方法覆盖。

pdf_{TeX}: 新的警告忽略参数 `\pdfsuppresswarningpagegroup`; 用来制造词间空白 (`interword space`) 的新命令: `\pdfinterwordspaceon`, `\pdfinterwordspaceoff`, `\pdffakespace`, 它们可以帮助 PDF 文本重新排版 (`reflowing`),

Lua_{TeX}: 对于字体载入和断字 (`hyphenation`) 有明显的变化和修正。最大的增加时一个新的引擎, `luajittex` 和它的相关变体 `texluajittex` 和 `texluajittextc`。它使用的是一个即时编译的 Lua 编译器 (在 *TUGboat* 文章 <https://tug.org/TUGboat/tb34-1/tb106scarso.pdf> 有详细介绍)。`luajittex` 还在

开发中，所以并没有在所有平台提供，也比 `luatex` 要不稳定许多。我们及其作者都不建议使用它，除非为了试验 Lua 代码的即时编译 (JIT) 这样特殊的目的。

XeTeX: 现在所有平台 (包括 Mac) 都支持所有的图像格式了。避免使用 Unicode 兼容性 decomposition fallback (但允许其他的变体); 为了与先前版本 XeTeX 的兼容, 优先于 Graphite 字体使用 OpenType 字体。

MetaPost: 支持了一个新的数字系统 `decimal` (十进制), 还包括一个配套的内部 `numberprecision`; 在 `plain.mp` 有 Knuth 对 `drawdot` 的新定义; SVG 和 PNG 输出的 bug 修正, 等等。

独立的 `pstopdf` ConTeXt 实用程序会在这个版本后被去除, 因为和系统同名程序有冲突。但你仍然可以通过 `mtxrun --script pstopdf` 命令来执行它。

`psutils` 被它新的维护者大幅更新了。使得许多很少使用的工具 (`fix*`, `getafm`, `psmerge`, `showchar`) 都只在 `scripts/` 目录中提供, 而不作为所有用户访问的应用程序 (如果以后发现这么做有问题, 我们还可以改进)。另外, 还加入了一个新脚本, `psjoin`。

MacTeX 这个 TeX Live 的重新发行版 (见第 3.1.2 节) 不再包括可选的、只在 Mac 下提供的 Latin Modern 和 TeX Gyre 字体包, 因为用户现在已经很容易在自己的系统下安装这些字体了。来自 ImageMagick 的 `convert` 程序也被去除了, 因为 TeX4ht (具体说来是 `tex4ht.env`) 现在直接使用 Ghostscript 了。

包含中文、日文和韩文的 `langcjk` 集合被拆分为独立的语言集合, 使得每个的大小更合理。

平台: 加入了 `x86_64-cygwin`, 去除了 `mips-irix`; Microsoft 不再支持 Windows XP, 所以我们的程序也可能在以后不支持这个系统。

9.1.12 2015

L^AT_EX 2_ε 现在默认包含了以前必须自己手工载入的 `fixltx2e` 包的内容, 所以现在手工载入这个包不起任何作用了。一个新的 `latexrelease` 包和其他的机制控制了实现的功能。L^AT_EX News #22 和 “L^AT_EX changes” 文档有更多的信息。顺带, `babel` 和 `psnfss` 包尽管是 L^AT_EX 的核心部分, 是独立维护的, 所以不受这些改变的影响 (而且应该仍然继续有用)。

L^AT_EX 2_ε 的内部现在包含了和 Unicode 相关的引擎配置 (包括哪些字符是字母、primitive 的命名等等), 这些配置以前是 TeX Live 的一部分。这个变化对用户应该是没有影响的; 有些底层的内部控制命令被改名或者去除了, 但行为应保持一致。

pdfTeX: 支持 JPEG Exif 和 JFIF; 就算在 `\pdfinclusionerrorlevel` 是负值时也不输出警告; 同步到 xpdf 3.04 版本。

LuaTeX: 加入用于扫描 token 的 `newtokenlib`; 对 `normal` 随机数生成器等的 bug 修正。

XeTeX: 图像处理的修正; 优先选择与 `xetex` 处在同一个目录的 `xdvipdfmx` 程序; 内部的 XDV 操作符有所变化。

MetaPost: 新的 `binary numbersystem`; 新的支持日语的 `upmpost` 和 `updvitomp` 程序, 类似 `up*tex`。

MacTeX: 更新了自带的 Ghostscript 包的 CJK 支持。TeX Distribution 选项面板支持了 Yosemite (macOS 10.10)。XeTeX 不再支持 `resource-fork font suitcase` 格式 (通常没有扩展名); `data-fork suitcase (.dfont)` 仍然是支持的。

基础设施: 重新实现了 `fmtutil` 脚本来按每个独立的目录树读取 `fmtutil.cnf` 文件, 类似 `updmap`。Web2C `mktex*` 脚本 (包括 `mktexlsr`, `mktexfm`, `mktexpk`) 现在优先选择自己所在目录的程序, 而不总选

用当前 PATH 里的。

平台: *-kfreebsd 被去除了, 因为 T_EX Live 现在在这个平台下通过系统自带的安装方式已经很容易获取了。

部分额外平台提供了定制二进制包 (<https://tug.org/texlive/custom-bin.html>)。此外, 为节省空间部分平台现在没有在 DVD 中提供, 但可以通过网络安装。

9.1.13 2016

LuaT_EX: 原生命令有大量改变, 包括重命名和去除, 还有一些节点结构的重整。这些修改在 Han Hagen 的“LuaT_EX 0.90 对于 PDF 的后端改变及更多” (<https://tug.org/TUGboat/tb37-1/tb115hagen-pdf.pdf>) 一文中有详细的总结。如果了解更多细节, 参见 LuaT_EX 手册, `texmf-dist/doc/luatex/base/luatex.pdf`。

Metafont: 新的实验性姊妹程序 MFlua 和 MFluajit, 为了试验性目标在 METAFONT 中集成 Lua。

MetaPost: Bug 修正和内部为了 MetaPost 2.0 做的准备。

SOURCE_DATE_EPOCH 支持所有除 LuaT_EX 以外的引擎 (下一个版本支持全部引擎), 原始的 `tex` 也故意不支持: 如果设置了 SOURCE_DATE_EPOCH 环境变量, 它的值会被用作 PDF 输出的时间戳。如果还设置了 SOURCE_DATE_EPOCH_TEX_PRIMITIVES, SOURCE_DATE_EPOCH 的值会被用于初始化 T_EX 的原生命令 `\year`, `\month`, `\day`, `\time`。pdfT_EX 手册有例子和细节。

pdfT_EX: 新的原生命令 `\pdfinfoomitdate`, `\pdftrailerid`, `\pdfsuppressptexinfo`, 用来控制每次会执行变化的输出值。这些功能只用于 PDF 输出, 不用于 DVI。

XeT_EX: 新的原生命令 `\XeTeXhyphenatablelength`, `\XeTeXgenerateactualtext`, `\XeTeXinterwordspaceshaping`, `\mdfivesum`; 字符分类限制提高到了 4096; DVI 版本提升。

其他实用工具:

- `gregorio` 是一个新的程序, 作为 `gregoriotex` 宏包的一部分用于 Gregorian 圣歌音乐的排版; 它缺省就被包括在了 `shell_escape_commands` 中。
- `upmendex` 是一个创建索引的程序, 基本上和 `makeindex` 兼容, 支持了 Unicode 排序, 以及其他的一些修改。
- `afm2tfm` 现在只将基于音调的高度调整上调; 新选项 `-a` 忽略所有调整。
- `ps2pk` 可以处理扩展 PK/GF 字体了。

MacT_EX: 去除了 T_EX Distribution Preference Pane; 它的功能现在在 TeX Live Utility 里提供了; 更新了捆绑的 GUI 应用; 给需要在 Ghostscript 中使用多种 CJK 字体的用户提供了新的 `cjk-gs-integrate` 脚本。

基础架构: 支持系统级别的 `tlmgr` 配置文件; 校验包的完整性; 如果有 GPG 还会校验网络更新的签名。安装程序和 `tlmgr` 都会作这些检查。如果没有 GPG 则跳过这一步。

平台: `alpha-linux` 和 `mipsel-linux` 被移除了。

9.1.14 2017

LuaT_EX: 更多 callback, 更多排版控制, 更多对内部机制的控制; 在一些平台增加了用来动态载入代码的 `ffi` 库。

pdf_T_EX: 去年的 SOURCE_DATE_EPOCH_TEX_PRIMITIVES 环境变量被改名为 FORCE_SOURCE_DATE, 没有功能改变。如果 \pdfpageattr token 列表包含了字符串 /MediaBox, 则会忽略输出默认的 /MediaBox。

Xe_T_EX: Unicode/OpenType 数学支持现在是基于 HarfBuzz 的 MATH table 支持实现的; 还有一些 bug 修正。

Dvips: 保证最后一次设置的 papersize special 是最后选中的纸张尺寸, 和 dvipdfmx 以及宏包希望的效果一致; -L0 选项 (L0 配置文件设置) 可以用来恢复以前的行为, 也就是第一次设置纸张尺寸的 special 才有效。

ep_T_EX, eup_T_EX: 来自 pdf_T_EX 新的原生命令 \pdfuniformdeviate, \pdfnormaldeviate, \pdfrandomseed, \pdfsetrandomseed, \pdfelapsedtime, \pdfresettimer。

Mac_T_EX: 从这一年开始, Mac_T_EX 只支持 Apple 还在发布安全更新的 macOS 版本, 使用的平台名称是 x86_64-darwin, 目前这包括 Yosemite, El Capitan, 和 Sierra (10.10 及以后的)。给更老版本的 macOS 用的二进制程序不会包含在 Mac_T_EX 里面, 但 _T_EX Live 里还会提供 (x86_64-darwinlegacy, i386-darwin, powerpc-darwin)。

基础架构: TEXMFLOCAL 树现在 (默认) 是在 TEXMFSYSCONFIG 和 TEXMFSYSVAR 之前搜索了; 这是为了更好地符合本地文件优先于系统文件的原则。另外, tlmgr 有一个新的 shell 模式, 进行交互和脚本使用, 和一个新的 conf auxtrees 功能来方便添加和删除额外的目录树。

updmap 和 fmtutil: 这些脚本如果在没有明确指定是系统模式 (updmap-sys, fmtutil-sys, 或者 -sys 选项), 还是用户模式 (updmap-user, fmtutil-user, 或者 -user 选项) 的时候会给出警告。这是为了减少常见的小心启用用户模式而导致系统更新无法生效。细节请参见 <https://tug.org/texlive/scripts-sys-user.html>。

install-tl: Mac 下的个人路径, 比如 TEXMFHOME, 现在默认设置为和 Mac_T_EX 一样的 (~/.Library/...)。增加了新选项 -init-from-profile 来以特定 profile 启动安装; 新命令 P 来直接保存一个 profile; 新的 profile 变量名 (但先前的名字也还支持)。

Sync_T_EX: 临时文件的名字现在被改为了 foo.synctex(busy), 而不是 foo.synctex.gz(busy) (不再有 .gz)。需要删除临时文件的前端和编译系统因此需要调整。

其他工具: texosquery-jre8 是一个新的跨平台程序, 用来在 _T_EX 文档中获取系统 locale 和其他 OS 信息; 它已经包含在了 shell_escape_commands 中, 可以在受限的 shell 环境中执行。(texosquery 也支持更老的 JRE 版本, 但不能在受限模式下启用, 因为 Oracle 由于安全问题不再支持那些版本了。)

平台: 见上述的 Mac_T_EX 条目。没有其他的变化。

9.1.15 2018

Kpathsea: 对于非系统目录缺省进行大小写不敏感的文件名匹配; 可以通过设置 texmf.cnf 或者将 texmf_casefold_search 环境变量设置为 0 来禁用。详细信息见 Kpathsea 手册 (<https://tug.org/kpathsea>)。

ep_T_EX, eup_T_EX: 新的原语 \epTeXversion。

Lua_T_EX: 准备在 2019 年升级到 Lua 5.3: 大部分平台下都提供了一个 luatex53 程序, 但必须被改名为 luatex 才能使用。或者使用 Con_T_EXt Garden (<https://wiki.contextgarden.net>) 的文件; 那里有更多的信息。

MetaPost: 对错误的路径方向、TFM 和 PNG 输出的修正。

pdfTeX: 允许位图字体使用 encoding vector; 当前目录不再被加入 PDF ID 的计算; 关于 `\pdfprimitive` 的 bug 修正及相关改变。

XeTeX: 在插入 PDF 图像文件时支持 `/Rotate`; 如果输出程序失败会返回非 0 退出码; 许多 UTF-8 和原语的修正。

MacTeX: 参见下面关于 macOS 版本支持的变化。除此之外, MacTeX 在 `/Applications/TeX/` 下安装的文件经过了重新组织使之更为清晰; 现在这个目录包含四个 GUI 程序 (BibDesk, LaTeXiT, TeX Live Utility, 和 TeXShop), 它的子目录则包含其余的实用程序和文档。

tlmgr: 新的 `tlshell` (Tcl/Tk) 和 `tlcockpit` (Java) 前端; JSON 输出; `uninstall` 现在可以替代 `remove` 使用; 新的 `print-platform-info` 选项。

平台支持:

- 去除: `armel-linux`, `powerpc-linux`。
- `x86_64-darwin` 支持 10.10–10.13 (Yosemite, El Capitan, Sierra, 和 High Sierra)。
- `x86_64-darwinlegacy` 支持 10.6–10.10 (尽管 10.10 最好使用 `x86_64-darwin`)。去除了所有对 10.5 (Leopard) 的支持, 也就是说 `powerpc-darwin` 和 `i386-darwin platforms` 被去除了。
- Windows: 不再支持 XP。

9.1.16 2019

Kpathsea: 更一致的括号展开和路径拆分; 用来替代 `.` 的 `TEXMFDOTDIR` 新变量, 以方便子目录搜索 (见 `texmf.cnf` 中的注释)。

epTeX, eupTeX: 新的原语 `\readpapersizespecial` 和 `\expanded`。

LuaTeX: 现在使用的是 Lua 5.3, 随之而来的是算术和接口的变化。自带的 `pplib` 用于读取 PDF 文件, 所以可以去掉对于 `poppler` 的依赖 (也就不再依赖 C++); Lua 接口有相应的变化。

MetaPost: `r-mpost` 命令被认作加了 `--restricted` 选项的别名, 并被加入默认提供的受限命令中。十进制和二进制模式的最小精度现在是 2。MPLib 不再提供二进制模式, 但独立的 MetaPost 中仍然提供。

pdfTeX: 新的原语 `\expanded`; 如果新的原语 `\pdfomitcharset` 的参数被设置为 1, 那么 PDF 输出会忽略 `/CharSet` 字符串, 因为不能保证它是正确的, 而 PDF/A-2 和 PDF/A-3 要求保证它的正确性。

XeTeX: 新的原语 `\expanded`, `\creationdate`, `\elapsedtime`, `\filedump`, `\filemoddate`, `\filesize`, `\resettimer`, `\normaldeviate`, `\uniformdeviate`, `\randomseed`; 扩展 `\Ucharcat` 来输出 active 的字符。

tlmgr: 支持用 `curl` 作为下载程序; 优先用 `lz4` 和 `gzip` 而不是 `xz` 做本地备份; 优先使用系统提供的压缩和下载程序, 而不是 TeX Live 自带的, 除非设置了 `TEXLIVE_PREFER_OWN` 环境变量。

`install-tl`: 新的 `-gui` 选项 (不带参数) 现在是 Windows 和 Mac 下的默认的选项, 它会使用一个新的 Tcl/TK 图形界面 (见第 1.3 节和第 3.1.6 节)。

实用程序:

- `cwebbin` (<https://ctan.org/pkg/cwebbin>) 现在是 TeX Live 中的 CWEB 实现, 支持更多的方言, 并包括构建 mini-index 使用的 `ctwill` 程序。
- `chkdvifont`: 报告 DVI 文件中字体的信息, 包括 `tfm/ofm`, `vf`, `gf`, `pk` 格式。

- `dvispc`: 让 DVI 文件的 `specials` 对页面无关。

MacTeX: `x86_64-darwin` 现在支持 10.12 或更新的版本 (Sierra, High Sierra, Mojave); `x86_64-darwinlegacy` 仍然支持 10.6 或更新版本。不再包含 Excalibur 拼写检查器, 因为它需要 32 位支持。

平台支持: 去除了 `sparc-solaris`。

9.1.17 2020

通用的更新:

- 所有 TeX 引擎, 包括 `tex` 的 `\input` 原语, 现在都支持以组分隔的文件名参数, 作为跟系统有关的扩展。标准的用空格或者标识符分隔的文件名的使用完全不受影响。按组分隔的参数一开始是 LuaTeX 先实现的; 现在它被推广到所有的引擎中了。ASCII 双引号字符 (") 一开始会被从文件名中去掉, 但在输入后不变。这个改变目前不会影响 LaTeX 的 `\input` 命令, 因为那是对标准 `\input` 原语重新定义的宏。
- `kpsewhich`, `tex`, `mf` 以及所有其他引擎的新选项 `--cnf-line, tex`, 以支持通过命令行指定任意配置文件。
- 今年和往年对多个引擎添加的许多原语是为了让所有引擎都拥有一系列共通的功能 (见 LaTeX News #31, <https://latex-project.org/news>)。

epTeX, eupTeX: 新的原语 `\Uchar`, `\Ucharcat`, `\current(x)spacingmode`, `\ifincsname`; 更新 `\fontchar??` 和 `\iffontchar`。只针对 eupTeX 的: `\currentcjktoken`。

LuaTeX: 与 HarfBuzz 库的集成, 作为新引擎 `luahtex` (被用于 `lualatex`) 和 `luajithbtex` 提供。新的原语: `\eTeXgluestretchorder`, `\eTeXglueshrinkorder`。

pdfTeX: 新的原语 `\pdfmajorversion`; 这个命令只改变 PDF 输出的版本号; 它对任何 PDF 内容没有影响。`\pdfximage` 等类似命令现在以和 `\openin` 相同的方式来搜索图像文件了。

pTeX: 新的 `\ifjfont`, `\iftfont` 原语, 它们也包含在 epTeX, upTeX, eupTeX 里。

XeTeX: 对 `\Umathchardef`, `\XeTeXinterchartoks`, `\pdfsavepos` 的修正。

Dvips: 位图字体的输出编码, 以更好的支持复制粘贴 (<https://tug.org/TUGboat/tb40-2/tb125rokicki-type3search.pdf>)。

MacTeX: MacTeX 和 `x86_64-darwin` 现在需要 10.13 或者更新的系统 (High Sierra, Mojave, 和 Catalina); `x86_64-darwinlegacy` 支持 10.6 和更新的系统。MacTeX 已经被公证 (notarized) 命令行程使用强化的运行时环境, 因为 Apple 现在要求安装包必须提供这些。BibDesk 和 TeX Live Utility 不再是 MacTeX 的一部分, 因为它们没有被公证, 但提供的 README 文件里列出了获取它们的链接。

tlmgr 和基础架构:

- 在软件包无法现在时, 自动重试一次。
- 新选项 `tlmgr check texmfdb`s, 用来检查每个目录树 `ls-R` 文件和 `!!` 配置的一致性。
- 软件包容器使用带版本号的文件名, 比如 `tlnet/archive/pkgname.rN.NN.tar.xz`; 对用户应该没有影响, 但对于分发来说会是一个明显的改变。
- `catalogue-date` 信息不在从 TeX Catalogue 获得了, 因为它通常和软件包更新没有关系。

9.1.18 2021

一般更新:

- Donald Knuth 对 T_EX 和 Metafont 进行的 2021 年调优更改已被整合 (<https://tug.org/TUGboat/tb42-1/tb130knuth-tuneup21.pdf>)。这些更改也可在 CTAN 上作为 knuth-dist 和 knuth-local 包提供。如预期的那样, 这些修复仅适用于罕见情况, 并不会影响实际操作中的行为。
- 除了原始 T_EX 外: 如果 \tracinglostchars 设置为 3 或更高, 缺失字符将导致错误, 而不仅仅是在日志文件中显示消息, 缺失的字符代码将以十六进制显示。
- 除了原始 T_EX 外: 一个新的整数参数 \tracingstacklevels, 如果为正值, 并且 \tracingmacros 也为正值, 将在每个相关日志行上输出一个前缀, 指示宏展开的深度 (例如, 深度为 2 时为 ~...)。此外, 宏记录将在深度 \geq 参数值时被截断。

Aleph: 基于 Aleph 的 L^AT_EX 格式 lamed 已被移除。aleph 二进制文件本身仍然包含在内并支持。

LuaT_EX:

- Lua 5.3.6。
- 用于 \tracingmacros 的嵌套级别回调, 作为新的 \tracingstacklevels 的通用变体。
- 将数学字符标记为保护, 以防止作为文本处理。
- 删除传统数学代码路径的宽度/ic 补偿。

MetaPost:

- 支持用于可重复输出的 SOURCE_DATE_EPOCH 环境变量。
- 避免在 mpto 中错误地结尾 %。
- 记录 -T 选项, 其他手册修复。
- 在二进制和十进制模式下更改 epsilon 的值, 以便 mp_solve_rising_cubic 函数按预期工作。

pdfT_EX:

- 新增原语 \pdfrunninglinkoff 和 \pdfrunninglinkon; 例如, 用于禁用在页眉和页脚中生成链接。
- 当 “\pdfendlink 结束于与 \pdfstartlink 不同的嵌套级别” 时发出警告而不是中止。
- 在 fnt 文件中转储 \pdfglyphtounicode 分配。
- 源代码: 删除 poppler 支持, 因为同步上游太难。在原生 T_EX Live 中, pdfT_EX 一直使用 libs/xpdf, 这是从 xpdf 中裁剪和调整的代码。

XeT_EX: 修复了数学字符间距问题。

Dvipdfmx:

- 现在默认安全调用 Ghostscript; 要覆盖此设置 (从而假设所有输入文件都是可信的), 请使用 -i dvipdfmx-unsafe.cfg。要使用 PSTricks 与 XeT_EX, 这是必需的, 如下所示:
xetex -output-driver="xdvipdfmx -i dvipdfmx-unsafe.cfg -q -E" ...
- 如果找不到图像文件, 则以错误状态退出。
- 扩展的颜色支持特殊语法。
- 用于操作 ExtGState 的特殊命令。
- 兼容性特殊命令 pdfcolorstack 和 pdffontattr。
- 实验性支持 dviluatex 的扩展 fnt_def。
- 支持虚拟字体的新特性以回退日文字体定义。

Dvips:

- 默认 PostScript 文档标题现在是输入文件的基本名称，可以通过新选项 `-title` 覆盖。
- 如果找不到 `.eps` 或其他图像文件，则以错误状态退出。
- 支持虚拟字体的新特性以回退日文字体定义。

MacTeX: MacTeX 及其新的二进制文件夹 `universal-darwin` 现在需要 macOS 10.14 或更高版本 (Mojave, Catalina 和 Big Sur); 不再存在 `x86_64-darwin` 二进制文件夹。通过 Unix `install-tl` 提供的 `x86_64-darwinlegacy` 二进制文件夹支持 10.6 及更新版本。

今年对 Macintosh 来说是重要的一年，因为 Apple 在 11 月推出了 ARM 机器，并将在多年内销售和支持 ARM 和 Intel 机器。`universal-darwin` 中的所有程序都具有 ARM 和 Intel 的可执行代码。两种二进制文件都从相同的源代码编译。

额外的程序 Ghostscript、LaTeXiT、TeX Live Utility 和 TeXShop 都是通用的，并且都签署了加固运行时，因此它们今年都包含在 MacTeX 中。

tlmgr 和基础设施:

- 仅保留主存储库的 `texlive.tlpdb` 的一个备份。
- 在系统和 Perl 版本之间实现更高的可移植性。
- `tlmgr info` 报告新的 `lcat-*` 和 `rcat-*` 字段，用于本地与远程目录数据。
- 子命令的全部日志记录移动到一个新的日志文件:
`texmf-var/web2c/tlmgr-commands.log`。

9.1.19 2022

一般更新:

- 新引擎 `hitex`，它输出自己的 HINT 格式，特别为在移动设备上阅读技术文档而设计。GNU/Linux、Windows 和 Android 的 HINT 查看器可以从 TeX Live 单独获取。
- `tangle`、`weave`: 支持可选的第三个参数来指定输出文件。
- Knuth 的程序 `twill` 现在包含在内，用于为原始 WEB 程序制作迷你索引。

跨引擎扩展 (原始 TeX、Aleph 和 hiTeX 除外):

- 新增原语 `\showstream`，用于将 `\show` 输出重定向到文件。
- 新增原语 `\partokenname` 和 `\partokencontext`，允许覆盖空行、`vbox` 结束等时发出的 `\par` 标记的名称。

epTeX, eupTeX:

- 新增原语: `\lastnodefont`、`\suppresslongerror`、`\suppressoutererror`、`\suppressmathparerror`。
- 现在可用 pdfTeX 扩展 `\adjust pre`。

LuaTeX:

- 支持 PDF 2.0 的结构化目标。
- PDF 2.0 的 PNG /Smask。
- `luahbtex` 的可变字体接口。
- 数学默认模式中不同的根式样式默认值。

- 可选地阻止选定的可分字符创建。
- 改进 TrueType 字体实现。
- 更高效的 `\fontdimen` 分配。
- 忽略只有本地 `par` 节点后跟方向同步节点的段落。

MetaPost: 无限宏展开的错误修复。

pdf \TeX :

- 支持 PDF 2.0 的结构化目标。
- 对于字母间隔字体, 如果指定了显式 `\fontdimen6`, 则使用它。
- 始终在行首发出警告。
- 对于具有自动调整字符间距 (`\pdfappendkern` 和 `\pdfprependkern`) 的字符, 仍然进行突出显示; 同样, 对隐式和显式连字符进行自动调整。

p \TeX 等:

- p \TeX 重大更新至 4.0.0, 以更好地支持当前 L \TeX 。
- 新增原语 `\ptexlineendmode` 和 `\toucs`。
- `\ucs` (以前在 `uptex`、`euptex` 中可用) 现在在 p \TeX 和 ep \TeX 中也可用。
- 区分八位字符和日文字符, 如 Hironori Kitagawa 在 TUGboat 文章中讨论的那样 (<https://tug.org/TUGboat/tb41-3/tb129kitagawa-char.pdf>)。

Xe \TeX : 新增包装脚本 `xetex-unsafe` 和 `xelatex-unsafe`, 用于简化调用需要 Xe \TeX 和 PSTricks 透明操作符的文档, 这本质上是不安全的 (直到 Ghostscript 中重新实现为止)。为了安全, 使用 Lua(L \TeX) \TeX 。

Dvipdfmx:

- 支持 PSTricks 而无需 `-dNOSAfer`, 除了透明度。
- `-r` 选项再次有效, 用于设置位图字体分辨率。

Dvips: 默认情况下, 不尝试自动媒体调整以旋转纸张尺寸; 新选项 `--landscaperotate` 重新启用。

upmendex: 实验性支持阿拉伯语和希伯来语; 改进的字符分类和语言支持。

Kpathsea: `kpsewhich -all` 返回的第一个路径现在与常规 (非全部) 搜索相同。

tlmgr 和基础设施:

- 默认使用 https 访问 mirror.ctan.org。
- 使用 `TEXMFROOT` 而不是 `SELFAUTOPARENT` 以便于重新定位。
- `install-tl`: 如果下载或安装失败, 将自动继续并稍后重试 (一次)。

Mac \TeX : Mac \TeX 及其二进制文件夹 `universal-darwin` 需要 macOS 10.14 或更高版本 (Mojave、Catalina、Big Sur、Monterey)。仅通过 Unix `install-tl` 提供的 `x86_64-darwinlegacy` 二进制文件夹支持 10.6 (Snow Leopard) 及更新版本。

平台: 今年 (2022 年) 没有平台支持的变化。但是, 明年的发行版 (2023 年) 我们计划将 Windows 二进制文件从 32 位切换到 64 位。不幸的是, 我们无法同时支持两者。

9.1.20 2023

Windows: 正如之前宣布的那样, $\text{T}_{\text{E}}\text{X}$ Live 现在包含 64 位 Windows 二进制文件, 而不是 32 位的。新目录名为 `bin/windows` (将 64 位二进制文件放入名为“32”的目录中似乎不合适)。我们知道这将给 Windows 用户带来额外的工作, 但似乎没有更好的替代方案。如果需要 32 位的 Windows 支持, 你需要安装 $\text{T}_{\text{E}}\text{X}$ Live 2022。从 Windows 11 开始, 基于 ARM 的 Windows 支持 64 位 AMD/Intel 的仿真, 因此可以使用当前的 $\text{T}_{\text{E}}\text{X}$ Live。然而, 基于 ARM 的 Windows 10 无法仿真 Intel/AMD 64 位, 因此也需要 32 位 $\text{T}_{\text{E}}\text{X}$ Live。请参阅单独的 $\text{T}_{\text{E}}\text{X}$ Live Windows 网页 (<https://tug.org/texlive/windows.html>)。

跨引擎扩展 (原始 $\text{T}_{\text{E}}\text{X}$ 和 $\text{e-}\text{T}_{\text{E}}\text{X}$ 除外): `\special` 后跟一个新关键字“`shipout`”, 将参数标记的扩展延迟到 `\shipout` 时进行, 就像非 `\immediate\write` 一样。

$\text{epT}_{\text{E}}\text{X}$ 、 $\text{eupT}_{\text{E}}\text{X}$:

- 不再构建“原始” $(\text{u})\text{ptex}$; $(\text{u})\text{ptex}$ 现在以 $\text{e}(\text{u})\text{ptex}$ 的兼容模式运行。 $\text{pT}_{\text{E}}\text{X}$ 工具也同样, 具体如下。
- 新增原语: `\tojis`、`\ptextracingfonts`、`\ptexfontname`。
- 对于 `\font`, 支持 JIS/UCS 的新语法。

$\text{LuaT}_{\text{E}}\text{X}$:

- 新的原语 `\variablefam` 允许数学字符保持其类别, 同时让族适应。
- 改进了从右到左的注释区域
- 上述跨引擎“延迟`\special`”。
- 在 2023 年 5 月发布的 $\text{LuaT}_{\text{E}}\text{X}$ 1.17.0 中: 默认禁用了 `socket` 库; 新的命令行选项 `--socket` 启用了它, 以及 `--shell-escape` (而不是 `--shell-restricted`)。此外, `mime` 库现在始终可用, 并且新的函数 `os.socketsleep` 和 `os.socketgettime` 也始终可用。最后, 修复了一些绕过各种安全功能的隐藏方法。

MetaPost: 修复了一些错误。`svg->dx` 和 `svg->dy` 现在是 `double` 类型, 以提高精度; 更新了 `mp_begin_iteration`; 修复了 `mplib` 中的内存泄漏。

$\text{pdfT}_{\text{E}}\text{X}$:

- 新增原语 `\pdfomitinfodict`, 完全省略 `/Info` 字典。
- 新增原语 `\pdfomitprocset`, 控制省略 `/ProcSet` 数组: 如果该参数为负, 则包含 `/ProcSet`; 如果该参数为 0 并且 `pdftex` 生成 PDF 1.x 输出, 则包含 `/ProcSet`。
- 使用 `\pdfinterwordspaceon` 时, 如果当前字体的编码在位置 32 有一个 `/space` 字符, 则使用该字符; 否则, 使用 (新) 默认字体 `pdftexspace` 的 `/space` 字符。可以使用新的原语 `\pdfspacefont` 覆盖该默认字体。同样的新过程用于 `\pdffakespace`。

$\text{pT}_{\text{E}}\text{X}$ 等:

- 如上所述, `ptex` 现在以兼容模式运行 `eptex`, 而不是单独构建。
- $\text{pT}_{\text{E}}\text{X}$ 工具 (`pbibtex`、`pdvitype`、`ppltotf`、`ptftopl`) 合并到相应的 $\text{upT}_{\text{E}}\text{X}$ 版本中, 并以兼容模式运行。

$\text{XeT}_{\text{E}}\text{X}$: 修复了 `\topskip` 和 `\splittopskip` 在 `\XeTeXupwardsmode` 激活时的计算; 上述跨引擎“延迟`\special`”。

Dvipdfmx : 新增选项 `--pdfm-str-utf8`, 用于创建 `pdfmark` 和/或书签。

$\text{BIBT}_{\text{E}}\text{X}$ u:

- 这个 BibTeX 变体与 BibTeX 大部分向上兼容, 具有更好的 (基于 Unicode 的) 多语言支持。它已经在 TeX Live 中存在多年。
- 今年, 增加了更多支持 CJK 语言的功能, 一些功能从日本的 (u)pibibtex 和其他程序扩展而来。

Kpathsea: 在类 Unix 平台上支持猜测输入文件编码, 如同在 Windows 上一样; 启用于 (e) p (la) tex、pibibtex、mendex。

tlmgr 和基础设施:

- 在 macOS 上默认使用文本界面。
- 首先安装核心包, 其他包重试一次。
- 对磁盘空间是否足够进行简单检查。

MacTeX:

- MacTeX 及其二进制文件夹 universal-darwin 需要 macOS 10.14 或更高版本 (Mojave、Catalina、Big Sur、Monterey、Ventura)。仅在 Unix install-tl 中可用的 x86_64-darwinlegacy 二进制文件夹支持 10.6 (Snow Leopard) 及更高版本。
- MacTeX 中的 GUI 包现在包含 hintview, 一个 macOS 查看器用于 HINT 文档 (由 hitex 和 hilatex 引擎为移动设备创建; 请参阅 HiTeX 网页, <https://hint.userweb.mwn.de/hint/hitex.html>)。GUI 包不再安装文档文件夹, 而是用一个简短的 READ ME 文件和一个关于 hintview 的页面取代。
- DVD 上的 Extras 文件夹中额外的 TeX 软件已被包含下载链接的文档取代。

平台:

- 如上所述, 新的 windows 二进制目录包含 64 位 Windows 二进制文件。
- bin/win32 二进制目录已删除, 因为我们无法同时支持 32 位和 64 位 Windows。
- i386-cygwin 二进制目录已删除, 因为 Cygwin 不再支持 i386。

9.1.21 2024

跨引擎变化 (原始 TeX、e-TeX 和 LuaTeX 除外):

- 避免因 \output 和 \write 大括号的恶意使用导致的崩溃。
- 微小的内循环优化。(这些更改均由 David Fuchs 完成。)

LuaTeX:

- Lua 级别的文件写入检查现在与 TeX 级别的检查类似。
- 如果设置了 Lua 值 texconfig.trace_extra_newline, 则在宏参数之间的日志中输出空行。在 2020-2023 年, 这种情况是默认的; 在此之前, 不会输出此类空行, 这是其他引擎的行为。
- 过去, Windows 平台的 texlua (而不是 luatex, 也不是其他平台) 内部会对参数执行通配符扩展。现在不再发生这种情况。

MetaPost:

- 现在包含用于区间算术的 MPFI 库; 使用 --numbersystem=interval 启用。

pdfTeX:

- 在大端系统上更可靠的文本提取。

- 增加对损坏的 pfb、fmt、pdf 文件的检查。
- 避免由于不寻常的 pdf 包含导致写入损坏的 pdf 文件。
- 手册转换为 L^AT_EX 并进行了彻底审查，但实质性更新较少。

epT_EX、eupT_EX:

- 新的简单编码转换工具 ptekf。
- eptex 不再单独构建; 它在 euptex 的遗留编码模式下运行。

XeT_EX:

- 支持文件名中超出基本多文种平面的 Unicode 字符。

Dvipdfmx:

- 以与 pdfT_EX 相同的方式计算时区偏移。
- 处理变体选择器以应对新的 Adobe-Japan1-UCS2 CMap。
- 在虚拟字体中为 (u)pT_EX 回退多字节字符到 OFM 字体。
- 支持 pdf:image 特殊命令中的 named 关键字。

Kpathsea:

- 支持扩展的安全文件名检查，同时允许在 TEXMF[SYS]VAR 下写入，适用于 Lua(L^A)T_EX; 新增函数和相应的 kpsewhich 选项。
- 支持新的变量 TEXMF_OUTPUT_DIRECTORY (与传统的 TEXMFOUTPUT 一起)，使子程序可以访问引擎调用中的 --output-directory 设置。

CWEB:

- 支持以 LuaT_EX 作为后端 T_EX 引擎。
- 修复了在无换行点的行上发生的缓冲区溢出问题。

tlmgr:

- 新动作 bug，帮助查找错误报告信息。
- 在所有平台上使用我们自己的根证书包，以更广泛地支持 LWP，特别是在 macOS 上。
- 检测远程仓库是否比本地安装更旧 (即过时的镜像)，并且不更新，以避免错误的包移除。
- 支持 ConT_EXt 纸张尺寸设置。
- 修剪镜像仓库 (主仓库和 tlcontrib) 的 tlpdb 文件备份。

平台:

- ISO 镜像再次包含所有平台的二进制文件。在过去的几年中，一些二进制集被修剪，但今年我们没有作为用户组福利制作实体 DVD，所以可以再次包含所有内容。而且，即使是最大限度地修剪的镜像也太大，无法适应单层 DVD，因此修剪对 2024 年的志愿刻录者没有好处。更多信息请访问 <https://tug.org/texcollection/>。
- Robert Alessi 在 <https://sr.ht/~ralessi/texlive-openbsd/> 提供了在 OpenBSD 上原生 T_EX Live 安装的二进制文件。

MacT_EX: 包含最新的 Ghostscript 10.03.0，所有四个 GUI 程序的更新，以及 (如往常一样) 所有新的 T_EX Live 更改。二进制文件仍然支持 macOS 10.14 (Mojave, 2018) 及更高版本。(如往常一样，更早的 Mac 版本由 x86_64-darwinlegacy 二进制文件支持，这些文件是 T_EX Live 的一部分，但不是 MacT_EX 的一部分。)

9.2 当前: 2025

跨引擎变化 (原始 T_EX 除外):

- 在所有格式 (包括纯 T_EX 和 L^AT_EX) 都生成 PDF-1.7, 之前是 1.5。若可能需要使用较旧版本, 特别是对于印刷公司, 则可以通过以下几种方式生成, 以 PDF-1.4 为例:
 - 在 T_EX 生成 PDF 后运行 `ps2pdf14`; 这个简单的包装程序是 Ghostscript 的一部分。其他外部软件也能完成同样的工作。
 - 使用支持 PDF 后端的 L^AT_EX: `\pdf_version_gset:n{1.4}`
 - pdfT_EX: `\pdfmajorversion=1 \pdfminorversion=4`
 - LuaT_EX: `\pdfvariable majorversion=1 \pdfvariable minorversion=4`
 - XeT_EX: `\special{pdf:majorversion:1}\special{pdf:minorversion:4}`
- 如果 T_EX 因为 `-halt-on-error` 选项退出, 默认行为是将帮助信息写入日志文件。
- 新的原语 `\ignoreprimitiverror`; 如果设置为 1, 则错误 “Infinite glue shrinkage found in box being split” 会变成警告, 因此程序的退出状态不会改变为失败。其他值保留供未来使用。(背景: <https://www.latex-project.org/news/latex2e-news/ltnews39.pdf>, 章节 “Enhancements to the new mark mechanism”)
- 缩放字体到 $\geq 2048\text{pt}$ 时, 现在会出现错误消息, 而不是 (未处理的) 计算溢出或悄悄改变用户的值。

LuaT_EX:

- LuaJIT 2.1.1736781742。
- 新的 `luaffi` 库, 支持 `aarch64`。
- 如果 `\tracinglostchars=4`, 在处理字形时会警告缺失字符, 而不是等到排版时才报告。如果 `\tracinglostchars=5`, 则将 “早期” 诊断转换为错误。
- 新的原语 `\mathemptydisplaymode`; 当设置为 1 时, 显示公式前的空段落将被忽略。默认情况下, 行间公式前包含方向节点的段落不会被忽略, 这是传统行为。
- 对 PNG 处理进行修改, 我们不再在掩码上设置属性。如果收到关于此问题的报告, 可以添加一个 `maskattr` 功能, 但应该考虑直接包含图像的正确 pdf, 因为它更为稳健。
- 在即时赋值中处理令牌组合器。
- 上述跨引擎更新。

pdfT_EX:

- 新的原语 `\pdfptexuseunderscore`, 用于在各种字典条目中使用 `_` 替代 `.`, 例如 `PTEX_Fullbanner` 替代 `PTEX.Fullbanner`。
- 不因为字体变化而省略真实空格。
- 增加检测过长行中的单词间空格的容差。
- 忽略嵌入图像中像素为 `1x1` 的微小分辨率, 而不是超过 `\maxdimen`。
- 上述跨引擎更新。

epT_EX、eupT_EX:

- 支持多个代码点的组合字符, Unicode 拉丁字符以及 16 位 OFM Level-0 新的 `kcatcodes:modifier(20)`, `latin_ucs(14)`。
- 上述跨引擎更新。

XeT_EX:

- 上述跨引擎更新。

Dvipdfmx:

- `ebb` 可执行文件现在是一个独立维护的脚本，而不是指向 `dvipdfmx` 的链接。
- 支持 upTeX 新的编码方式，通过虚拟字体合并字符。
- 修复了包含超过 127 个字形的嵌入 TrueType 字体的问题等。
- 修复了多次包含图像的问题。

CWEB:

- 新选项 `+F` 用于 `cweave` 和 `ctwill` 处理复合语句。
- 同步 pdf 书签页面与目录。
- 集成 Martin Ruckert 的解决方案以加快索引排序。
- 支持宏 `\9` 用于排序章节名。
- 改进了 `ctwill-proofsor` 中的排序算法。
- 输出章节名的 `ctwill` 格式。

Dvips:

- 支持 upTeX 新的编码方式，通过虚拟字体合并字符。

Kpathsea:

- 当 `kpsewhich` 找不到给定文件时，如果指定了多个文件进行搜索，将输出空行。

tlmgr:

- `tlmgr info`: 新选项 `--only-files`，仅报告给定包的文件。
- `tlmgr search`: 新选项 `--json`。

平台:

- 现在 `x86_64-linux` 二进制文件是在 Alma Linux 8 上构建的，因为 CentOS 7 的支持 (包括安全更新) 已于 2024 年 6 月 30 日结束。对此我们深感遗憾，但别无选择。旧系统的二进制文件仍然可以使用 (见 <https://tug.org/texlive/custom-bin.html>)。
- TeX Live 现在提供文件 `scripts-targets.lst`，包含符号链接脚本的名称和目标，方便分发版使用。它位于源目录 `Build/source/texk/texlive/linked_scripts` 中。长期以来的文件 `scripts.lst`，仅包含目标，仍然被维护。

Windows:

- `uninstall-windows` 和 `runscript.tlu` 中的 Visual Basic 用法已被 Powershell 取代，因为 Windows 11 起 VB 变为可选。

MacTeX: 今年，MacTeX-2025 支持 macOS 10.14 (Mojave, 2018) 及更高版本的 Arm 和 Intel 处理器。它安装了:

- TeX Live 2025。
- Ghostscript-10.04.0。
- `libgs` 和 `mutool`，用于 `dvisvgm`。以前需要 X11 才能支持 `dvisvgm`，但 2025 版本取消了这一要求，`dvisvgm` 在有无 X11 的情况下均可正常工作。
- HintView 2.0，支持理解 TeX Live 2025 中最新的 `hitex` 添加的颜色扩展。
- Asymptote，现在需要 macOS 11 (Big Sur, 2020) 或更高版本。
- TeXShop 5.49，支持 Apple 最新的 Apple Intelligence 特性，包括写作工具。

9.3 未来

我们打算继续发布新的 T_EX Live 版本, 并希望提供更多的文档、更多的程序、更完善和更全面检查的宏和字体树以及其他与 T_EX 相关的内容。所有这些工作都是由志愿者在业余时间完成的, 而且总有更多的工作要做。请参阅<https://tug.org/texlive/contribute.html>。

请将纠正、建议和帮助的提议发送到:

tex-live@tug.org (您无需订阅即可发布)
<https://tug.org/texlive>

祝你使用 T_EX 愉快!

10 翻译说明

这里对简体中文版本《T_EX Live 指南》, 即本文档中遵循的翻译惯例作一简要说明:

- package, 视上下文, 有时翻译为软件包, 有时翻译为宏包。
- format file, 即 T_EX 程序一般都会预载入的 .fmt 文件。本文档中翻译为格式文件。
- scheme, 本文档中译为 (安装) 方案。
- collection, 本文档中译为 (软件) 集合。
- 本文档中有时对原文没有采用逐字逐句的对比翻译, 而是总括其意思, 转换为更易为中文 T_EX 用户习惯的表达方式。
- architecture/platform, 是意思比较相近的词, 基本上是只某种 CPU 和对应这个 CPU 的操作系统。比如 i386-linux。本文里翻译为架构、平台、体系结构等等。
- binary, 二进制文件, 其实就是说可执行程序文件和库文件了。

2007 年的简体中文版本由 Jiang Jiang, Jinsong Zhao, Yue Wang, Helin Gai 翻译。其中 Jinsong Zhao 负责 Windows 部分的翻译, Yue Wang 和 Helin Gai 进行了校对, Jiang Jiang 则负责其余的翻译和统稿。

2008 年的简体中文版本由 Jiang Jiang, Yue Wang 和 Jinsong Zhao 翻译。

2009 年的简体中文版本由 Jiang Jiang 和 Jinsong Zhao 翻译。

2010 到 2020 年的简体中文版本都由 Jiang Jiang 翻译。

2024 年至今的简体中文版本由 Ran Wang (王然) 翻译。