

TEX Live 指南

TEX Live 2009

Karl Berry 编写

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

2009 年 10 月

目录

1 简介	3
1.1 TEX Live 与 TEX Collection	3
1.2 操作系统支持	3
1.3 TEX Live 的基本安装	3
1.4 获得帮助	4
2 TEX Live 概览	4
2.1 TEX Live, proTEXt, MacTEX 的大集合: TEX Collection	4
2.2 TEX Live 的顶层目录	5
2.3 预定义的 texmf 目录树概览	5
2.4 TEX 的扩展版本	6
2.5 TEX Live 中其他值得一提的程序	6
2.6 TEX Live 中的字体	7
3 安装	7
3.1 启动安装程序	7
3.1.1 Unix	7
3.1.2 Mac OS X	8
3.1.3 Windows	8
3.1.4 Cygwin	8
3.1.5 文本界面安装程序	8
3.1.6 专家图形界面安装程序	9
3.1.7 简化的向导安装程序	9
3.2 执行安装程序	9
3.2.1 二进制系统菜单 (只对 Unix 适用)	9
3.2.2 选择要安装的组件	10
3.2.3 目录	10
3.2.4 选项	11
3.2.5 设置从 DVD 运行 (只对文本模式适用)	13
3.3 install-tl 命令行选项	13

3.3.1	-repository 参数	14
3.4	安装后的操作	14
3.4.1	Windows	14
3.4.2	如果创建了符号链接	14
3.4.3	Unix 下的环境变量	14
3.4.4	环境变量的全局配置	15
3.4.5	XeTeX 的字体配置	15
3.4.6	如果从 DVD 运行	15
3.4.7	ConTeXt Mark IV	15
3.4.8	集成本地与个人宏文件	16
3.4.9	集成第三方字体	16
3.5	测试安装是否成功	16
3.6	其他可下载软件的链接	17
4	网络安装	18
5	在 DVD 或 USB 上最具可移植性地运行 TeX Live	18
6	tlmgr: 管理你的安装	19
6.1	图形界面模式的 tlmgr	19
6.2	tlmgr 命令行使用示例	20
7	有关 Windows 平台的说明	21
7.1	针对 Windows 的特征	21
7.2	Windows 上附加的软件	22
7.3	User Profile 目录相当于主目录	22
7.4	Windows 注册表	23
7.5	Windows 权限	23
8	Web2C 用户指南	23
8.1	Kpathsea 路径搜索	24
8.1.1	路径的来源	25
8.1.2	配置文件	25
8.1.3	路径展开	25
8.1.4	默认展开	26
8.1.5	大括号展开	26
8.1.6	子目录展开	26
8.1.7	特殊字符与其意义: 简要说明	26
8.2	文件名数据库	27
8.2.1	文件名数据库	27
8.2.2	kpsewhich: 独立的路径搜索	27
8.2.3	使用举例	28
8.2.4	调试操作	29
8.3	运行时选项	31

1 简介	3
9 致谢	32
10 发行历史	33
10.1 过去	33
10.1.1 2003	33
10.1.2 2004	34
10.1.3 2005	35
10.1.4 2006–2007	36
10.1.5 2008	36
10.2 现状	37
10.3 未来	37
11 翻译说明	37

1 简介

1.1 T_EX Live 与 T_EX Collection

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、Mac OS X 和 Windows 系统下的一套发行版。

你可以直接下载 T_EX Live，也可以在 T_EX 用户组织给会员分发的 T_EX Collection DVD 中找到。第 2.1 节中，简要地介绍了 DVD 的内容。这两套发行版都是用户组织共同协作完成的。这篇文档，主要介绍 T_EX Live 本身。

T_EX Live 包括了 T_EX, L^AT_EX 2_ε, ConT_EXt, METAFONT, MetaPost, BibT_EX 等许多可执行程序；种类繁多的宏包、字体和文档，并支持世界上许多不同的语言。

文档末尾的第 10 节 (第 33 页) 介绍了这一版 T_EX Live 的重要改变。

1.2 操作系统支持

T_EX Live 为包括 Mac OS X 在内的多种基于 Unix 的操作系统提供了可执行文件，也有 Cygwin 下的可执行文件。它还包含了源代码，可供在没有提供可执行文件的平台上编译安装。

至于 Windows，T_EX Live 仅支持 Windows 2000 或后续版本。不再支持 Windows 9x, ME 和 NT。因为这一改变，Windows 支持相对于 Unix 支持系统而言不再需要很多的特殊对待。我们没有包含 64 位的 Windows 可执行文件，不过 32 位的可执行文件也能 64 位的系统上正常运行。

除了 T_EX Live 以外，Windows 和 Mac OS X 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

你可以使用 DVD 方式或者网络方式来安装 T_EX Live。通过网络的安装程序本身非常小，它可以从网上下载所有的你所要求的软件包。网络安装程序对仅使用 T_EX Live 一小部分的用户来说非常适宜。

DVD 安装程序可以把 T_EX Live 安装到你的本地磁盘上，也可以直接从 DVD 上运行 T_EX Live (其实如果你的系统支持挂载 DVD 镜像，你也可以从 DVD 镜像上运行)。安装方法将在下面的章节介绍，这里提供一个快速入门：

- 安装脚本的名称是 `install-tl`。它可以在指定了 `-gui=wizard` 选项的情况下以“向导”模式 (Windows 下的默认模式) 工作，或指定 `-gui=text` 选项以文本模式 (其它系统下默认模式) 工作，还有一个专家 GUI 模式可以通过 `-gui=perltk` 选项启用。
- 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 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读者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 <http://tug.org/interest.html#vendors> 上有一份销售商的列表。)

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*，<http://tug.org/begin.html> 这个网页提供了这个系统的简短介绍。

***T_EX* FAQ** 这套庞大的 *T_EX* FAQ 对各种各样的问题——从最基础到最晦涩的——都给予了简明的回答，它在 *T_EX Live* 的 [texmf-dist/doc/generic/FAQ-en/html/index.html](http://tug.org/texmf-dist/doc/generic/FAQ-en/html/index.html)，也可以在 <http://www.tex.ac.uk/faq> 网站上找到。有问题时请先看看这里能否找到解答。

***T_EX* Catalogue** 如果你在寻找某个特定的宏包、字体、程序等等，*T_EX* Catalogue 就是你首先该找的地方。这里是所有 *T_EX* 相关内容的一个巨大集合。参见 <http://www.ctan.org/help/Catalogue/>。

***T_EX* 网上资源** <http://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。

支持信息的归档 最重要的两个技术支持论坛是 Usenet 的新闻组 `news:comp.text.tex` 和邮件列表 `texhax@tug.org`。它们的内容归档中有多年以来的提问和回答供你搜索。你可以分别从 <http://groups.google.com/groups?group=comp.text.tex> 和 <http://tug.org/mail-archives/texhax> 进行查询。当然，一般性的搜索方式，比如用 <http://google.com> 找找，总没有坏处。

提问 如果你还是找不到答案，就可以通过 Google 或者你的新闻组阅读器在 `comp.text.tex` 上提问，或者发送邮件到 `texhax@tug.org`。不过，在提问之前请一定先阅读 FAQ 上的这一条：<http://www.tex.ac.uk/cgi-bin/texfaq2html?label=askquestion>，它能提高你获得回答的可能性。

***T_EX* Live 技术支持** 如果你需要报告 bug，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` 参数来运行程序都一般能得到用来报告其 bug 的地址。

另一方面，你也不妨帮助其他有问题的朋友，`comp.text.tex` 与 `texhax` 都是对所有人开放的，请尽管参与进去，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* 的内容，以及包含 *T_EX Live* 的 *T_EX* Collection。

2.1 *T_EX* Live, pro*T_EX*t, Mac*T_EX* 的大集合：*T_EX* Collection

T_EX Collection 的 DVD 包含了以下内容：

***T_EX* Live** 是一个完整的 *T_EX* 系统，它可以直接在光盘上运行，也可以安装在本地磁盘上。它的主页在 <http://tug.org/texlive/>。

Mac*T_EX* 在 *T_EX Live* 的基础上增加了原生的 Mac OS X 的安装程序和一些其它的 Mac 应用程序。它的主页在 <http://tug.org/mactex/>。

pro*T_EX*t 是 Windows 下的 MiK*T_EX* 发行版的一个增强版本。pro*T_EX*t 在 MiK*T_EX* 基础上增加了一些工具，简化了安装。它完全独立于 *T_EX Live*，有其自己的安装步骤。pro*T_EX*t 的主页在 <http://tug.org/protext>。

CTAN 一份 CTAN 仓库的快照 (<http://www.ctan.org>)。

`texmf-extra` 一个提供额外的配套软件包的目录。

CTAN, `protext` 和 `texmf-extra` 并不一定遵循 \TeX Live 的版权协议, 因此在分发或修改时要格外地小心。

2.2 \TeX Live 的顶层目录

这里是 \TeX Live 发行版顶层目录的一个简短的列表和描述。注意在 \TeX Collection DVD 中, 整个 \TeX Live 目录结构都存放于 `texlive` 子目录下, 而非光盘的顶层目录下。

`bin` \TeX 系统程序, 按平台组织。

`readme.html` 网页, 提供了多种语言的简介和有用的链接。

`readme-*.dir` \TeX Live 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 Web2C \TeX 和 METAFONT 发行版。

`texmf` 见下文的 `TEXMFMAIN`。

`texmf-dist` 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, 以及一些对 Windows 的特殊支持。

上述目录之外, 安装脚本和 (多种语言的) `README` 文件也存放在发行版的顶层目录下。

至于文档, 顶层目录下的 `doc.html` 文件中提供的完整的链接会有帮助。程序的文档 (手册, `man page`, `Info` 文件等) 在 `texmf/doc` 目录下, 因为这些程序本身是属于 `texmf` 目录的。 \TeX 宏包与格式文件的文档则放在 `texmf-dist/doc` 目录。但不管放在哪个地方, 你都可以使用 `texdoc` 程序来寻找这些文档。

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

- 简体中文: `texmf/doc/texlive/texlive-zh-cn`
- 捷克 / 斯洛伐克语: `texmf/doc/texlive/texlive-cz`
- 英语: `texmf/doc/texlive/texlive-en`
- 法语: `texmf/doc/texlive/texlive-fr`
- 德语: `texmf/doc/texlive/texlive-de`
- 波兰语: `texmf/doc/texlive/texlive-pl`
- 俄语: `texmf/doc/texlive/texlive-ru`

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 \TeX 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。

`TEXMFMAIN` 存储系统核心部件的目录树, 包括配置文件、辅助脚本和程序文档。

`TEXMFDIST` 存储主要的宏包、字体等等。

`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、字体的目录。

`TEXMFHOME` 给用户存放它们自己独立安装的宏包、字体等等。这个变量根据不同的用户选择不同的主目录。

`TEXMFCONFIG` 给 `texconfig`、`updmap`、和 `fmtutil` 这些程序存储修改过的配置文件。默认在 `TEXMFHOME` 目录下。

`TEXMFSYSCONFIG` 给 `texconfig-sys`、`updmap-sys` 和 `fmtutil-sys` 这些程序存储修改过的文件。

`TEXMFVAR` 这个目录是给 `texconfig`、`updmap` 和 `fmtutil` 存储 (缓存) 格式文件、生成 `map` 文件这类运行时数据的。默认在 `TEXMFHOME` 目录下。

`TEXMFSYSVAR` 给 `texconfig-sys`、`updmap-sys` 和 `fmtutil-sys` 还有 `tlmgr` 这几个命令存储、缓存运行时使用的格式文件和生成的 `map` 文件。

默认的目录结构:

全系统根目录 可以包含多个 \TeX Live 版本:

```

2008 上一个版本。
2009 当前版本。
    bin
        i386-linux GNU/Linux 二进制文件
        ...
        universal-darwin Mac OS X 二进制文件
        win32 Windows 二进制文件
    texmf      这是 TEXMFMAIN。
    texmf-dist TEXMFDIST
    texmf-var  TEXMFSYSVAR
    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%)

```

.texlive2008 给上个版本的, 个人生成和配置的数据。
.texlive2009 给这个版本的, 个人生成和配置的数据。
    texmf-var    TEXMFVAR
    texmf-config TEXMFCONFIG
    texmf TEXMFHOME 个人的宏包文件, 等等。 等等。

```

2.4 \TeX 的扩展版本

原始的 Knuth \TeX 本身的开发已经冻结了, 仅仅修改除去发现的极其少量的错误。它在 \TeX Live 中仍然作为 `tex` 程序出现, 在可见的未来也仍然如此。 \TeX Live 包括了一些建立在 \TeX 基础上的扩展程序:

ε - \TeX 为 \TeX 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 \TeX -X \TeX T 扩展模式。在默认模式下, ε - \TeX 是与原始的 \TeX 100% 兼容的。参见 texmf-dist/doc/etex/base/etex_man.pdf。

pdf \TeX 在 ε - \TeX 扩展的基础上构建, 在 DVI 输出之外增加对 PDF 输出的支持, 以及许多其他的扩展。这是针对 `etex`, `latex` 或 `pdflatex` 这些格式使用的缺省程序。可以在 texmf/doc/pdftex/manual/samplepdf/samplepdf.tex 找到展示部分功能的例子。

Lua \TeX 是 pdf \TeX 指定的后继者, 而且对 pdf \TeX 保持大部分 (但不是完全地) 向下兼容。它也希望包含 Aleph (见后) 的功能, 尽管在技术上未必与它兼容。它内置的 Lua 语言解释器 (<http://www.lua.org>) 为许多棘手的 \TeX 问题提供了优雅解决方案。当以 `texlua` 命令执行时, 它就像一个标准的 Lua 解释器一样工作, 所以, Lua \TeX 在 \TeX Live 中也被作为 Lua 程序的解释器。见 <http://www.luatex.org> 和 texmf-dist/doc/luatex/luatexref-t.pdf。

Xe \TeX 通过第三方库, 增加对 Unicode 输入文本和 OpenType 字体的支持, 能够直接使用系统字体。参见 <http://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。它同时还支持所谓的 ‘ Ω Translation Processes’ (OTP), 用于对任意输入进行复杂的变换操作。Omega 现在已经不作为单独的程序出现在 \TeX Live 中了; 改为只支持 Aleph:

Aleph 将 Ω 与 ε - \TeX 扩展合并到一起得到的。在 texmf-dist/doc/aleph/base 可以找到一些简短的文档。

2.5 \TeX Live 中其他值得一提的程序

这里是在 \TeX Live 中其他的一些常用程序:

`bibtex` 参考文献支持。

`makeindex`, `xindy` 索引支持。

dvips 将 DVI 转换为 PostScript。

xdvi X Window System 下的 DVI 阅读器。

dvilj HP LaserJet 系列打印机的 DVI 驱动。

dviconcat, dviselect 从 DVI 文件中复制和粘贴页面。

dvipdfmx 将 DVI 转换为 PDF, 是 (前面提到过的) pdfTeX 的一套替换方案。参见 ps4pdf 和 pdftricks 软件包以了解其他的方案。

psselect, psnup, ... PostScript 实用程序。

texexec, texmfstart ConTeXt 和 PDF 处理工具。

tex4ht TeX 到 HTML (还有 XML 等其他格式) 的转换器。

2.6 TeX Live 中的字体

TeX Live 中包括了很多的高质量矢量字体, 参见 <http://tug.org/fonts> 和 texmf-dist/doc/fonts/free-math-fonts-survey。

3 安装

3.1 启动安装程序

首先请找来一张 TeX Collection 的 DVD, 或者下载 TeX Live 的网络安装程序, 然后找到这个安装脚本: Unix 下是 `install-tl`, Windows 下是 `install-tl.bat`。

网络安装程序: 在 CTAN 的 `systems/texlive/tlnet` 目录下可以下载, <http://mirror.ctan.org/systems/texlive/tlnet> 这个地址能自动将你导向一个附近的、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.bat` 就会出现在 `install-tl` 子目录中。

TeX Collection DVD: 打开 `texlive` 这个子目录。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 TeX 用户组织 (推荐这么做, 参见 <http://tug.org/usergroups.html>) 或是单独购买 (通过 <http://tug.org/store>), 又或者是自己从 ISO 镜像刻录。

参见 <http://tug.org/texlive/acquire.html> 以了解更多关于获取这个软件的信息和方法。

下面的章节介绍更详细地介绍了安装程序的启动。

3.1.1 Unix

`install-tl` 是一个 Perl 脚本。在 Unix 兼容的系统下启动它最简单的方法是这样的:

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

(或者你可以直接运行 `perl /path/to/installer/install-tl`, 或者 `./install-tl`, 如果这个文件有可执行属性, 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的大小才能在一屏内显示完整的文本安装程序界面 (图 1)。

要在专家 GUI 模式下安装 (见图 2; 你需要 Perl/Tk 模块), 使用:

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

要列出所有这些选项:

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

关于 Unix 下权限的警告： 在安装过程中， $\text{T}_{\text{E}}\text{X}$ Live 安装程序将会遵照你的 `umask` 行事。所以如果你需要让你的安装能给其他用户使用，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 并没有包含所有运行 $\text{T}_{\text{E}}\text{X}$ Live 安装程序所必须的程序，详见第 3.1.4 节。

3.1.2 Mac OS X

如第 2.1 节提到的，我们给 Mac OS X 准备了一套独立的发行版，叫做 Mac $\text{T}_{\text{E}}\text{X}$ (<http://tug.org/mactex>)。我们推荐使用原生的 Mac $\text{T}_{\text{E}}\text{X}$ 安装程序，而不是 $\text{T}_{\text{E}}\text{X}$ Live 自带的那个，因为原生的安装程序做了一些针对 Mac 的调整，尤其方便在多个 Mac OS X 下的 $\text{T}_{\text{E}}\text{X}$ 发行版 (Mac $\text{T}_{\text{E}}\text{X}$, gw $\text{T}_{\text{E}}\text{X}$, Fink, MacPorts, ...) 之间切换。

Mac $\text{T}_{\text{E}}\text{X}$ 是严格依赖 $\text{T}_{\text{E}}\text{X}$ Live 构建的，所以主 $\text{T}_{\text{E}}\text{X}$ 树也是完全一致的。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

如果你使用的是网络安装程序，又或者 DVD 安装程序无法自动启动了，请双击 `install-tl.bat`。如果你需要更多定制选项，比如要选择特定的包集合，请改为运行 `install-tl-advanced.bat`。

你也可以从命令行提示符下启动安装程序。下面 `>` 表示的就是提示符，用户输入用 **bold** 体表示。如果你正在安装程序目录下，只需要运行：

```
> install-tl
```

或者你也可以通过绝对路径来运行，比如：

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

这是对 $\text{T}_{\text{E}}\text{X}$ Collection DVD 而言的，假定 D: 是光驱。图 3 展示了向导安装程序，它是 Windows 下的默认形式。

要在文本模式下安装，使用：

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

要列出所有可用的选项：

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

3.1.4 Cygwin

$\text{T}_{\text{E}}\text{X}$ Live 安装程序只支持 Cygwin 1.7。在开始安装之前，请先使用 Cygwin 的 `setup.exe` 程序安装 `perl` 和 `wget` 软件包，如果你还没装过。此外还推荐安装下列软件包：

- `fontconfig` [Xe $\text{T}_{\text{E}}\text{X}$ 需要]
- `ghostscript` [各种实用工具需要]
- `libXaw7` [xdvi 需要]
- `ncurses` [给安装程序提供“清屏”命令]

3.1.5 文本界面安装程序

图 1 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。

这是一个完全用命令行操作的安装程序，完全不支持光标移动。所以你无法在多选框或者输入框之间用 `Tab` 上下切换，只能在提示符下输入特定的字符 (大小写敏感的) 然后按下 `Enter`，整个终端窗口就会把新的内容刷新出来。

文本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。

```

Installing TeX Live 2009 from: ...
Platform: i386-linux => 'Intel x86 with GNU/Linux'
Distribution: live (uncompressed)
...
Detected platform: Intel x86 with GNU/Linux

<B> binary systems: 1 out of 14

<S> Installation scheme (scheme-full)
    83 collections out of 84, disk space required: 1882 MB

Customizing installation scheme:
  <C> standard collections
  <L> language collections

<D> directories:
  TEXDIR (the main TeX directory):
    /usr/local/texlive/2009
  TEXMFLOCAL (directory for site-wide local files):
    /usr/local/texlive/texmf-local
  TEXMFSYSVAR (directory for variable and automatically generated data):
    /usr/local/texlive/2009/texmf-var
  TEXMFYSYSCONFIG (directory for local config):
    /usr/local/texlive/2009/texmf-config
  TEXMFHOME (directory for user-specific files):
    ~/texmf

<O> options:
  [ ] use letter size instead of A4 by default
  [X] create all format files
  [X] install macro/font doc tree
  [X] install macro/font source tree
  [ ] create symlinks to standard directories

<V> set up for running from DVD

Other actions:
  <I> start installation to hard disk
  <H> help
  <Q> quit

```

图 1: 文本安装程序主界面 (GNU/Linux)

3.1.6 专家图形界面安装程序

图 2 展示了 GNU/Linux 下的专家图形界面安装程序。除使用了按钮和菜单以外，这个安装程序和文本模式的 (图 1) 没什么区别。

这个模式可以通过下面的命令手工启动:

```
> install-tl -gui=perlTk
```

3.1.7 简化的向导安装程序

在 Windows 下，缺省会使用我们所支持的最简单的安装方法，也就是“向导”安装程序。它会把所有东西都装上，不提任何问题。如果你需要定制安装，请运行其他的安装程序。

这个模式可以通过下面的命令手工启动:

```
> install-tl -gui=wizard
```

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。

3.2.1 二进制系统菜单 (只对 Unix 适用)

图 4 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，你可以选择安装其他架构的二进制程序。这对你要将 TeX 树共享在异构网络上的情况比较有用，又或者用于双启动的系统。

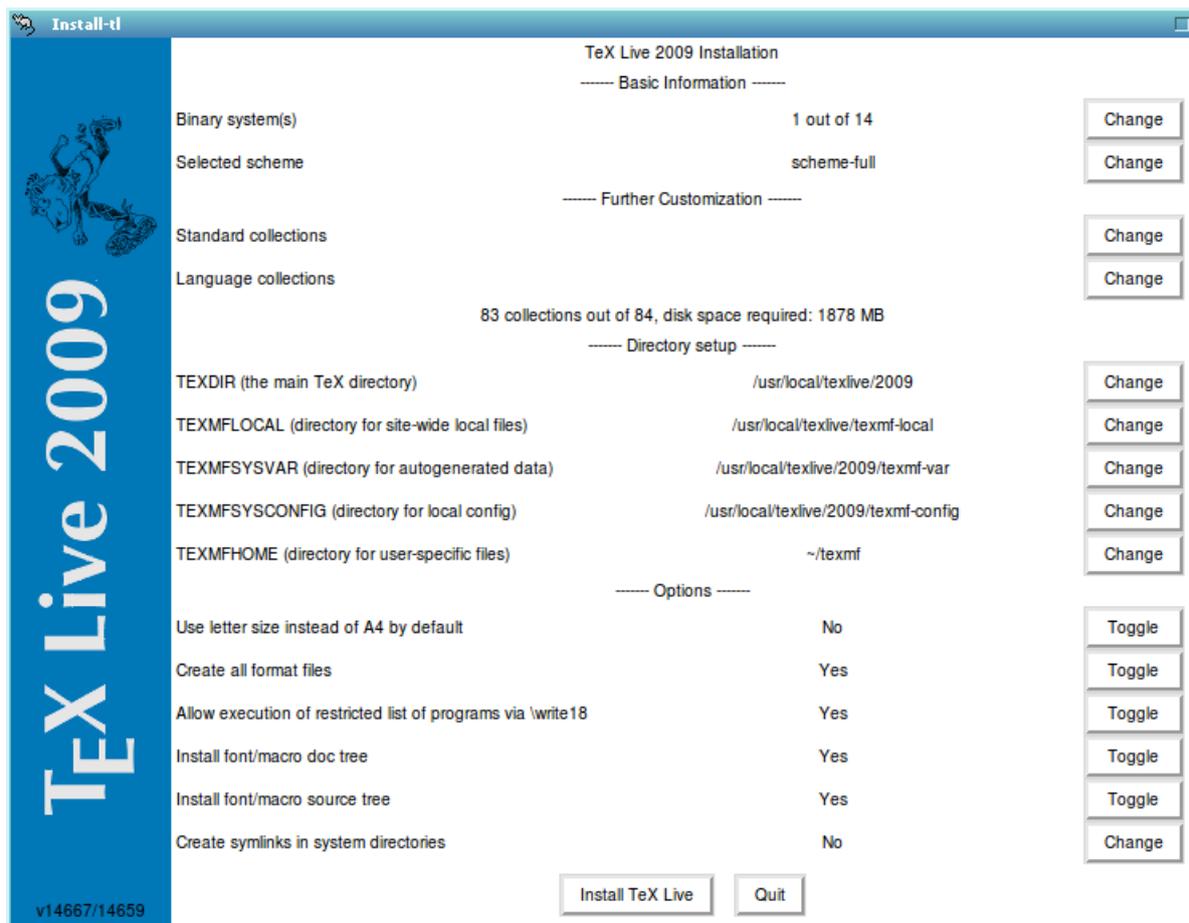


图 2: 专家模式 GUI 安装程序界面 (GNU/Linux)

3.2.2 选择要安装的组件

图 5 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，不过你也可以给小点的系统选择 `basic` 方案，为测试选用 `minimal` 方案，又或者是 `medium` 或 `teTeX` 来选择介乎其间的方案。还有许多特殊或者专门针对特定国家的方案。

你可以使用 ‘standard collections’ 与 ‘language collections’ 菜单来详细选择安装方案。(图 6 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集合，而一个集合又包含了一到多个软件包，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 `tlmgr` 程序 (参见第 6 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 5 页。默认的 `TEXDIR` 路径则是 Windows 下的 `%SystemDrive%\texlive\2009` 和 Unix 下的 `/usr/local/texlive/2009`。

更改默认值的主要原因大概是你可能没有默认位置的写权限。虽然要安装 TeX Live 不需要是管理员或者 root 用户，但你至少得对安装的目的目录有写权限。

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/2009’。

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。(你可能希望用一个类似 `/usr/local/texlive-cur` 这样的名字作为指向当前版本的符号链接，这样的目录名就和版本

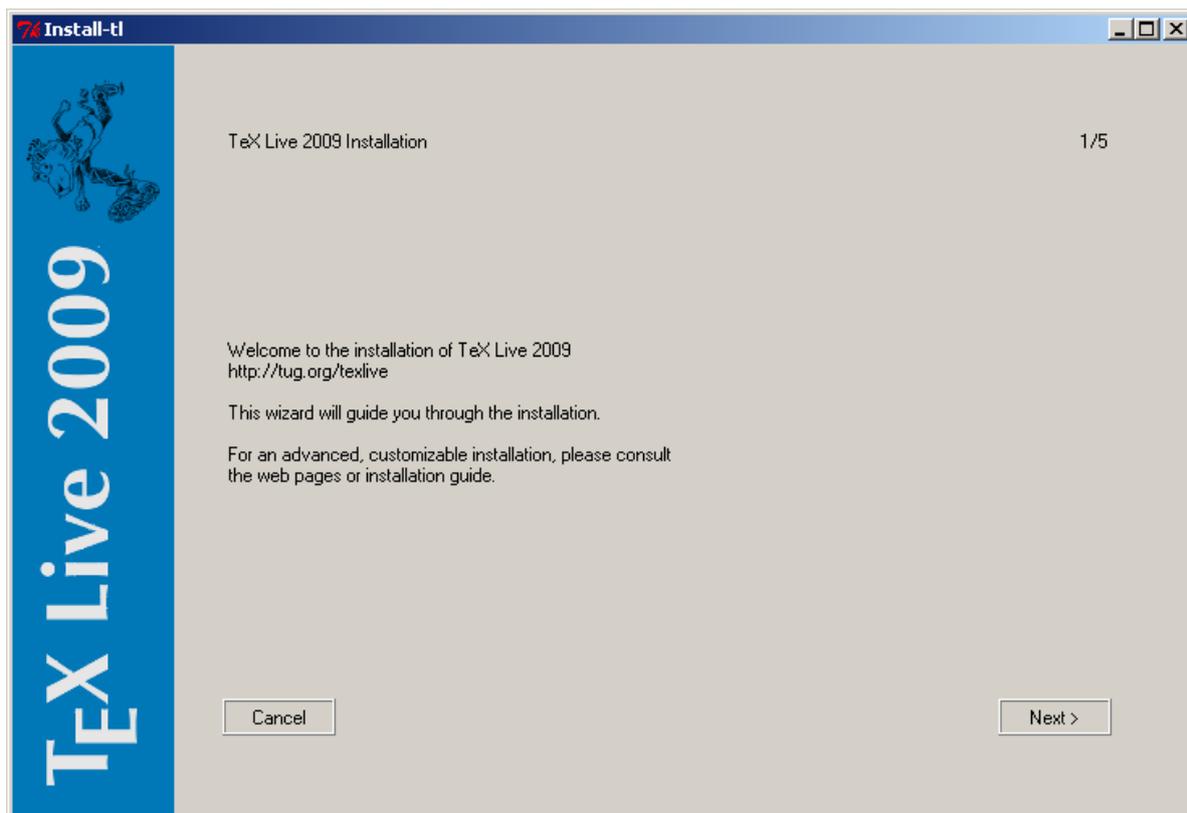


图 3: 向导模式安装程序界面 (Windows)

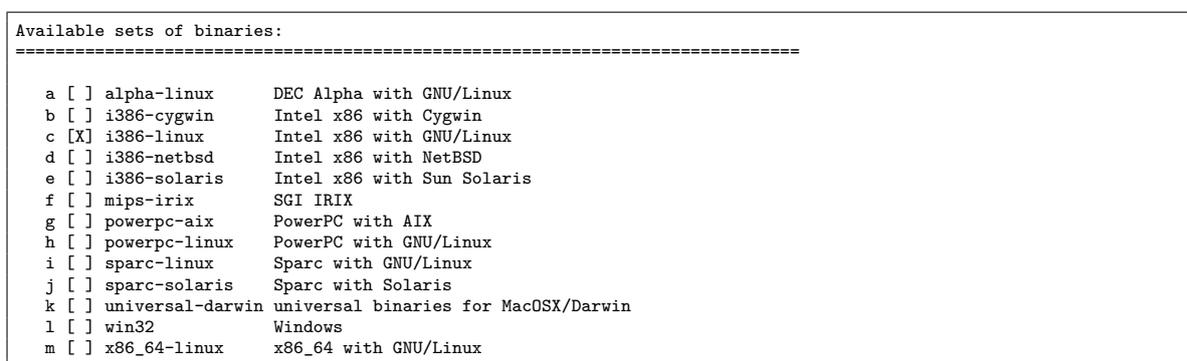


图 4: Binaries (二进制程序) 菜单

无关了，你只要在新版本出来直接更新符号链接即可。)

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

`TEXMFHOME` 是推荐用来存放个人宏文件和软件包的目录。其缺省值是 `~/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 `TeX` 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%`。

3.2.4 选项

图 7 显示了文本模式的选项菜单。关于这个菜单：

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

create format files: 虽然创建不必要的格式文件会浪费一点时间，也会多占一些磁盘空间，但我们还是建议现在保持这个选项的选定状态，因为如果这次不生成，下次用到的时候格式文件就会在

```

Select a scheme:
=====
a [ ] basic scheme
b [ ] ConTeXt scheme
c [X] full scheme (everything)
d [ ] GUST TeX Live scheme
e [ ] GUTenberg TeX Live scheme
f [ ] medium scheme (plain, latex, recommended packages, some languages)
g [ ] minimal scheme (plain only)
h [ ] Omega scheme
i [ ] teTeX scheme (more than medium, but nowhere near full)
j [ ] XML scheme
k [ ] custom selection of collections

```

图 5: Scheme (安装方案) 菜单

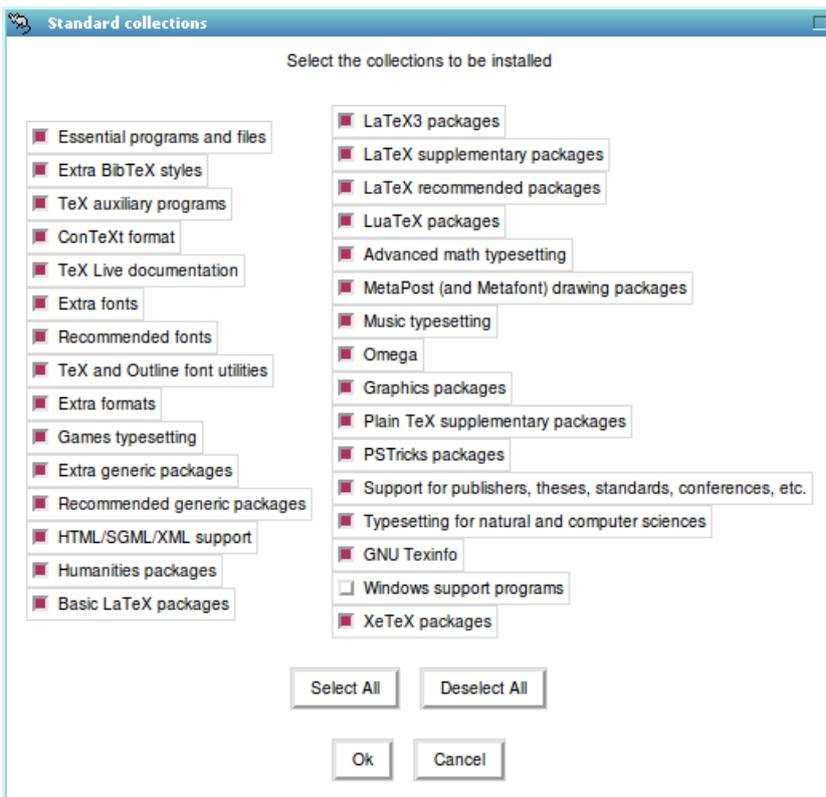


图 6: Collections (集合) 菜单

用户个人的 `TEXMFVAR` 目录树内生成。这样每次二进制文件或者断字模式更新的时候，这些格式文件也得不到更新，所以可能会导致它们的不兼容。

install font/macro ... tree: 这些选项允许你忽略下载安装大部分宏包中的文档和源代码文件。不建议使用。

create symlinks in standard directories (只对 Unix 有效): 这个选项可以省下设定环境变量的步骤。如果没有选择它，就必须把 `TEX Live` 的对应目录添加到 `PATH`, `MANPATH` 和 `INFOPATH` 中。如

```

<P> use letter size instead of A4 by default: [ ]
<F> create 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:

```

图 7: Options 菜单 (Unix)

果要创建符号链接，你需要对这些目标目录的写权限。我们强烈建议不要不要这个命令来覆盖现有的 $\text{T}_{\text{E}}\text{X}$ 系统，它主要是为了在用户已知的标准目录中创建符号链接设计的，这些目录并不包含任何 $\text{T}_{\text{E}}\text{X}$ 文件。

如果所有的设置已经齐备，你就可以按下 ‘I’ 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。

3.2.5 设置从 DVD 运行 (只对文本模式适用)

按 ‘V’ 来选择这个选项。它将主菜单改变为如图 8 的界面。

```

===== TeX Live installation procedure =====
...
<D> directories:
  TEXDIRW (Writable root):
    !! default location: /usr/local/texlive/2009
    !! is not writable, please select a different one!
  TEXMFLOCAL (directory for site-wide local files):
    /usr/local/texlive/texmf-local
  TEXMFYSVAR (directory for variable and automatically generated data):
    /usr/local/texlive/2009/texmf-var
  TEXMFYSCONFIG (directory for local config):
    /usr/local/texlive/2009/texmf-config
  TEXMFHOME (directory for user-specific files):
    ~/texmf

<O> options:
  [ ] use letter size instead of A4 by default
  [X] create all format files

<V> set up for installing to hard disk

Other actions:
<I> start installation for running from DVD
<H> help
<Q> quit

```

图 8: 启用 ‘from DVD’ 集合后的主菜单

注意变化之处：所有关于安装内容的选项都消失了，而目录这部分的设置现在是根据 `TEXDIRW` (或者一个可写的根目录) 来设置的。创建符号链接的选项也没有了。

这样安装程序仍然会创建一些目录和配置文件，但并不会把 `texmf`, `texmf-dist` 目录复制到本地硬盘上。

Unix 系统下的安装后配置步骤要稍微复杂一些，因为这样目录布局就和缺省情况有所区别了，参见第 3.4 节。

这个选项在 GUI 安装程序里没有，不过在 Unix 和 Windows 下都能用。Windows 用户需要从命令行选项来启动这个安装程序，参见第 3.3 节。

第 5 节描述了一套更具可移植性地运行 $\text{T}_{\text{E}}\text{X}$ Live 的方法，它不需要对系统配置做任何改动，但也做不了任何配置。

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。它需要安装了 Perl/Tk 模块 (<http://tug.org/texlive/distro.html#perlTk>)；如果找不到 Perl/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，就算在 Windows 下也是如此。如果你需要 ‘from DVD’ 安装方式就需要用到这个选项，因为这个方式在 GUI 模式下没有提供。

- lang *LL* 指定安装程序界面的语言，使用两个字符的语言代码 *LL*。目前支持的语言有：英语 (*en*, 默认值), 德语 (*de*), 法语 (*fr*), 荷兰语 (*nl*), 波兰语 (*pl*), 斯洛文尼亚语 (*sl*) 和越南语 (*vi*)。安装程序会尝试自己判断出合适的语言，如果判断出的语言没有支持就会使用英语替代。
- profile *profile* 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。这个选项可以让安装程序重用现成的这种文件，这样你就可以在后续系统里进行批量安装了，保证和第一次安装用的选项完全一样。
- repository *url-or-directory* 指定作为来源的软件包仓库，参见下文。

3.3.1 -repository 参数

默认的软件包安装位置是由 <http://mirror.ctan.org> 自动选择的 CTAN 镜像。

如果你需要自己指定，地址可以是以 `ftp:`, `http:` 或 `file:/` 起始的 URL，或者本地路径。(如果给定了 `http:` 或 `ftp:` 地址，其末尾的 `/` 字符和末尾的 `/tlpkg` 这段都会被忽略。)

比如你可以选择这样的 CTAN 镜像：<http://ctan.example.org/tex-archive/systems/texlive/tlnet/>，当然你应该把 `ctan.example.org` 替换为具体镜像的域名和特定的顶层 CTAN 路径 (<http://ctan.org/mirrors>) 维护了一个 CTAN 的镜像列表)。

如果给定的地址在本地磁盘上 (或者是路径或者是 `file:/` 开头的 URL)，就会使用指定路径下 `archive` 子目录中的压缩文件进行安装。

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 Windows

在 Windows 下安装程序会完成所有的操作。

3.4.2 如果创建了符号链接

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。

3.4.3 Unix 下的环境变量

你应该将自己使用的平台下二进制程序的目录加入搜索路径中。每个支持的平台都在 `TEXDIR/bin` 下有自己的子目录。参见图 4 以了解所有的这些子目录和它们对应的平台。

如果希望 `man` 和 `info` 能够找到，你还可以把文档的 `man` 和 `Info` 目录加入其对应的搜索路径中。在添加 `PATH` 后，手册页面可以被自动找到。

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

```
PATH=/usr/local/texlive/2009/bin/i386-linux:$PATH; export PATH
MANPATH=/usr/local/texlive/2009/texmf/doc/man:$MANPATH; export MANPATH
INFOPATH=/usr/local/texlive/2009/texmf/doc/info:$INFOPATH; export INFOPATH
```

对于 `csh` 或者 `tcsh`，需要修改的文件通常是 `$HOME/.cshrc`，而应该添加的内容类似：

```
setenv PATH /usr/local/texlive/2009/bin/i386-linux:$PATH
setenv MANPATH /usr/local/texlive/2009/texmf/doc/man:$MANPATH
setenv INFOPATH /usr/local/texlive/2009/texmf/doc/info:$INFOPATH
```

如果你已经在你的配置文件里写过了这样的路径设置，那就只需要把 TeX Live 的这些目录加进去就行了。

3.4.4 环境变量的全局配置

如果你希望在整个系统范围内修改这些变量，或这样系统新增的用户自动继承这些变量，就得自寻方法了，因为不同的系统之间这方面的配置差异太大。

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/2009/bin/i386-linux \
            /usr/local/texlive/2009/texmf/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。

我们还在每个 (Unix) 程序路径下创建了一个 `man` 符号链接。因为有些 `man` 程序，比如 Mac OS X 标准的 `man` 就能够自动通过这个链接找到对应的手册页，这样你就不必手工设置手册页路径了。

3.4.5 XeTeX 的字体配置

如果你在 Unix 兼容的系统中安装了 `xetex` 软件包，需要把系统配置一番 XeTeX 才能找到随 TeX Live 安装的那些字体。为了进行配置，`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。

要在整个系统中使用 TeX Live 的字体 (假定你有足够的权限)，请依照下面的步骤来做：

1. 将 `texlive-fontconfig.conf` 文件复制到 `/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，可以用下面的步骤将 TeX Live 字体提供给你自己，作为独立的 XeTeX 用户：

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

3.4.6 如果从 DVD 运行

通常来说 TeX Live 程序都会寻找一个叫做 `texmf.cnf` 的文件来获知所有的 TeX 树位置。它会在一系列和其本身位置相关的地方寻找这个文件。不过如果直接在 DVD 上运行，这套规则就不适用了：因为 DVD 是只读的，而某些路径只能在安装时确定下来并写入这个文件，所以这个文件没法放在 DVD 上，只能另置一处。所以可能会需要定义 `TEXMFCNF` 来告知 TeX Live 程序该从哪儿找到这个 `texmf.cnf` 文件。当然，如前所述 环境变量还是要改的。

在安装结束时，安装程序会输出一段告诉你如何设置 `TEXMFCNF` 的信息，如果你忽略了话，它应该是 `$TEXMFYSVAR/web2c`，默认情况下就是 `/usr/local/texlive/2009/texmf-var/web2c`，你需要在 `bash` 下使用：

```
TEXMFCNF=/usr/local/texlive/2009/texmf-var/web2c; export TEXMFCNF
```

或者 `[t]csh` 下使用：

```
setenv TEXMFCNF /usr/local/texlive/2009/texmf-var/web2c
```

如果你希望在自己的系统中运行 TeX Live 但磁盘空间不足，这个选项是很有用的。但要是你希望构建一套完全“便携”的、自包含的 TeX Live 系统，比如放在 USB 盘上，参见第 5 节。

3.4.7 ConTeXt Mark IV

‘老版’的 ConTeXt 正常就应该能工作。新的 ‘Mark IV’ ConTeXt 则需要一些手动设置。参见 http://wiki.contextgarden.net/Running_Mark_IV。

3.4.8 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 `C:\Program Files\texlive\texmf-local`）这个目录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。这些目录应该在各个 T_EX Live 版本之间共享，每个版本都能看到其内容。因此不应该把 TEXMFLOCAL 改得和主 T_EX Live 目录差别太大，否则新的版本出来你又得再改。

对于这两个目录树而言，文件都应该放到合适的子目录中，参见 <http://tug.org/tds> 或者 `texmf/web2c/texmf.cnf` 文件。比如一个 L^AT_EX 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，否则新增的文件就无法找到。你可以使用 `mktxlsr` 命令或者 `tlmgr` GUI 模式下，configuration 选项卡中的 ‘Reinit file database’ 按钮来刷新它。

3.4.9 集成第三方字体

不幸的是，这是一个非常混乱的问题。除非你愿意深入 T_EX 安装的细节，否则请不要涉足这个领域。别忘了事先看看我们附带的字体，参见第 2.6 节。

XeT_EX 是一套可行的替代方案（参见第 2.4 节），它能让你使用操作系统的字体而不必将它安装到 T_EX 中。

如果你非得这么做，参见 <http://tug.org/fonts/fontinstall.html>，这是我们对整个过程最好的描述。

3.5 测试安装是否成功

在完成你所需要的 T_EX 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。

这个小节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 Mac OS X 和 Windows 下你更可能是使用图形界面运行这些测试的，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 3.1415926 (TeX Live 2009)
kpathsea version 5.0.0
Copyright 2009 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 14 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：

```
> latex sample2e.tex
This is pdfTeX, Version 3.1415926-1.40.10 (TeX Live 2009)
...
Output written on sample2e.dvi (3 pages, 7484 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 29 页的“调试操作”一节以了解更多信息。）

3. 即时预览结果：

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

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。) `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。

4. 创建用于打印或显示的 PostScript 文件:

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

5. 创建 PDF 文件而非 DVI, 这里采用直接处理 .tex 文件输出 PDF 的方式:

```
> pdflatex sample2e.tex
```

6. 预览 PDF 文件:

```
> gv sample2e.pdf
```

或:

```
> xpdf sample2e.pdf
```

`gv` 和 `xpdf` 现在都不包含在 T_EX Live 中, 你必须单独安装它们。请分别参阅 <http://www.gnu.org/software/gv> 和 <http://www.foolabs.com/xpdf>。(还有许多其他的 PDF 查看器。) Windows 下我们推荐 Sumatra PDF (<http://blog.kowalczyk.info/software/sumatrapdf>)。

7. 除 `sample2e.tex` 外可能会对你有用的标准测试文件:

`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 (proofsheet) 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`’。

你可以用与我们处理 `sample2e.tex` 文件时相同的方式来处理这些文件。

8. 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 3.1415926...
```

```
...
```

```
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 XeT_EX 才能找到 T_EX Live 自带的字体。参见第 3.4.5 节。

3.6 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 <http://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 <http://www.cs.wisc.edu/~ghost/>

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

ImageMagick <http://www.imagemagick.com>, 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。这里列出了一些 (部分是 Windows 才有的)。

- GNU Emacs 在 Windows 下的原生版本在 <http://www.gnu.org/software/emacs/windows/nitemacs.html>。
- Emacs 的 AucTeX 包在 TeX Live DVD 的 `tlpkg/support` 目录；它的主页在 <http://www.gnu.org/software/auctex/>。
- LEd 在 <http://www.ctan.org/support/LEd> 提供。
- SciTE 在 <http://www.scintilla.org/SciTE.html> 提供。
- Texmaker 是自由软件，在 <http://pascal.brachet.club.fr/texmaker/> 提供。
- TeXnicCenter 是自由软件，在 <http://www.texniccenter.org> 提供，也随 proTeXt 发行版附带。
- TeXworks 是自由软件，在 <http://tug.org/texworks> 提供，也作为 TeX Live 的一部分在 Windows 和 Mac OS X 下被安装。
- Vim 是自由软件，在 <http://www.vim.org> 提供。
- WinShell 在 <http://www.winshell.de> 提供。
- WinEdt 是共享软件，在 <http://tug.org/winedt> 或 <http://www.winedt.com> 提供。

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

4 网络安装

TeX Live 的设计可以使它在不同的用户间共享，甚至可以在网络上不同的系统间共享。在标准的目录结构下，不需要配置固定的绝对路径：TeX Live 程序所需要的文件都能通过都在这些程序自身的相对路径找到。你可以在 `$TEXMFMAIN/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：

```
TEXMFMAIN = $SELFAUTOPARENT/texmf
...
TEXMFLOCAL = $SELFAUTOPARENT/./texmf-local
```

这就意味着，其它的系统或用户只需要把 TeX 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 TeX 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，可以在 <http://tug.org/texlive/w32client.html> 下载到一个叫做 `w32client` 的示例网络安装脚本。它可以为局域网上的 TeX Live 安装产生配置文件和菜单快捷方式。它还会注册一个叫 `w32unclient` 的卸载程序，在同一个 zip 文件中。参见它的网页以了解更多信息。

5 在 DVD 或 USB 上最具可移植性地运行 TeX Live

在第 3.2.5 节中介绍的“从 DVD 运行”选项可以使发行版在你的系统上很好地运行，但是如果你在别人的机器上用非管理员帐号工作，那你就可能希望使用对系统本身的影响最小的运行方式。

在 TeX Live DVD 的根目录，或者 TeX Collection DVD 的 `texlive` 子目录中，有一个为 Unix 系统提供的脚本 `tl-portable` 和一个为 Windows 系统提供的批处理文件 `tl-portable.bat`。它们能开启一个设置好环境变量的终端或命令提示符，在其中就能直接运行 DVD 上的 TeX Live。

头一次执行时会花点时间在 `~/tlportable2009` 目录下生成一些文件，此后就可以在很短的时间内启动了。

而此时系统的其它程序并不知道 TeX Live 的存在，如果你需要让你的文本编辑器使用这个 TeX Live，可以再打开一个这样的 `tl-portable` 会话来使用该编辑器。

你还能通过 `tl-portable` 在 USB 驱动器上运行 TeX Live。在这种情况下，请（至少）把所有的顶层目录下的所有文件和 `bin`, `texmf`, `texmf-dist` 和 `tlpkg` 下的所有内容复制到 USB 驱动器上。这会花一定的时间。如果你把它复制到一个 FAT32 格式的 USB 驱动器上，请保证复制操作是跟踪（dereference）所有的符号链接到实际数据文件的（`cp -L`）。运行时都会查找并使用驱动器上的 `texmf-local` 目录。

然后你就可以在驱动器的根目录下像上面那样运行 `tl-portable` 了。此时脚本能判断出这个驱动器是可写的，并把产生的文件写到这里。为了方便起见（比如你想把它提供给别人），你也可以把驱动器上的最终内容刻录回到 DVD 上。

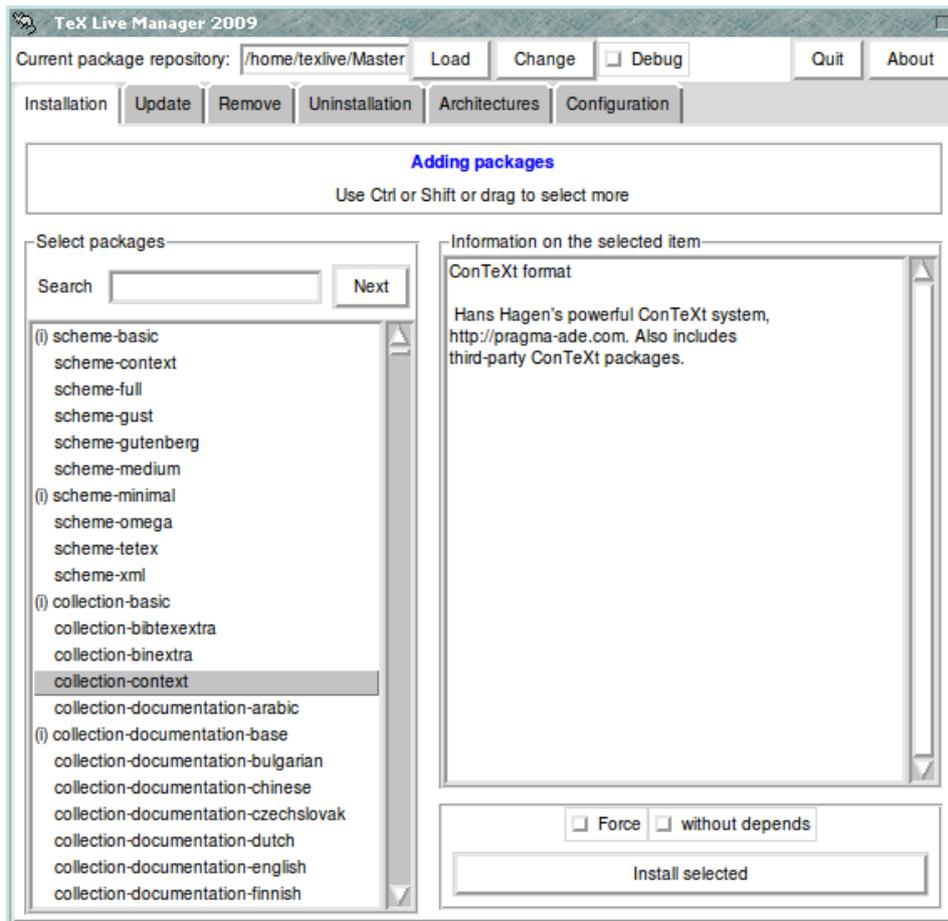


图 9: 图形界面模式运行的 `tlmgr` 程序。安装包 / 集合 / 方案仅在按下 ‘Load’ 按钮后显示。

6 tlmgr: 管理你的安装

TeX Live 包含一个叫 `tlmgr` 的程序，它可以用来管理安装后的系统。像 `updmap`, `fmtutil` 和 `texconfig` 这些程序仍然包含在发行版中，而且在将来也不会剔除，但是 `tlmgr` 是一个优选的界面。它的功能包括：

- 列出方案 (scheme)，集合和安装包；
- 安装、升级、备份、恢复、卸载软件包，并且能自动计算依赖关系；
- 查找和列出软件包；
- 列出、添加和删除不同架构的可执行文件；
- 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)。

警告：`tlmgr` 并不为在 DVD 中运行设计，也从未做过相关的测试。

6.1 图形界面模式的 `tlmgr`

`tlmgr` 可以用以下命令以图形模式启动：

```
> tlmgr -gui
```

Windows 下可以通过开始菜单：开始，程序，TeX Live 2009, TeX Live Manager。在假定安装包源是有效且可及的情况下，在点击 ‘Load’ 以后，它会列出所有可获取的或已安装 (前面用 ‘(i)’ 表示) 的软件包。

图 10 展示了配置选项卡。

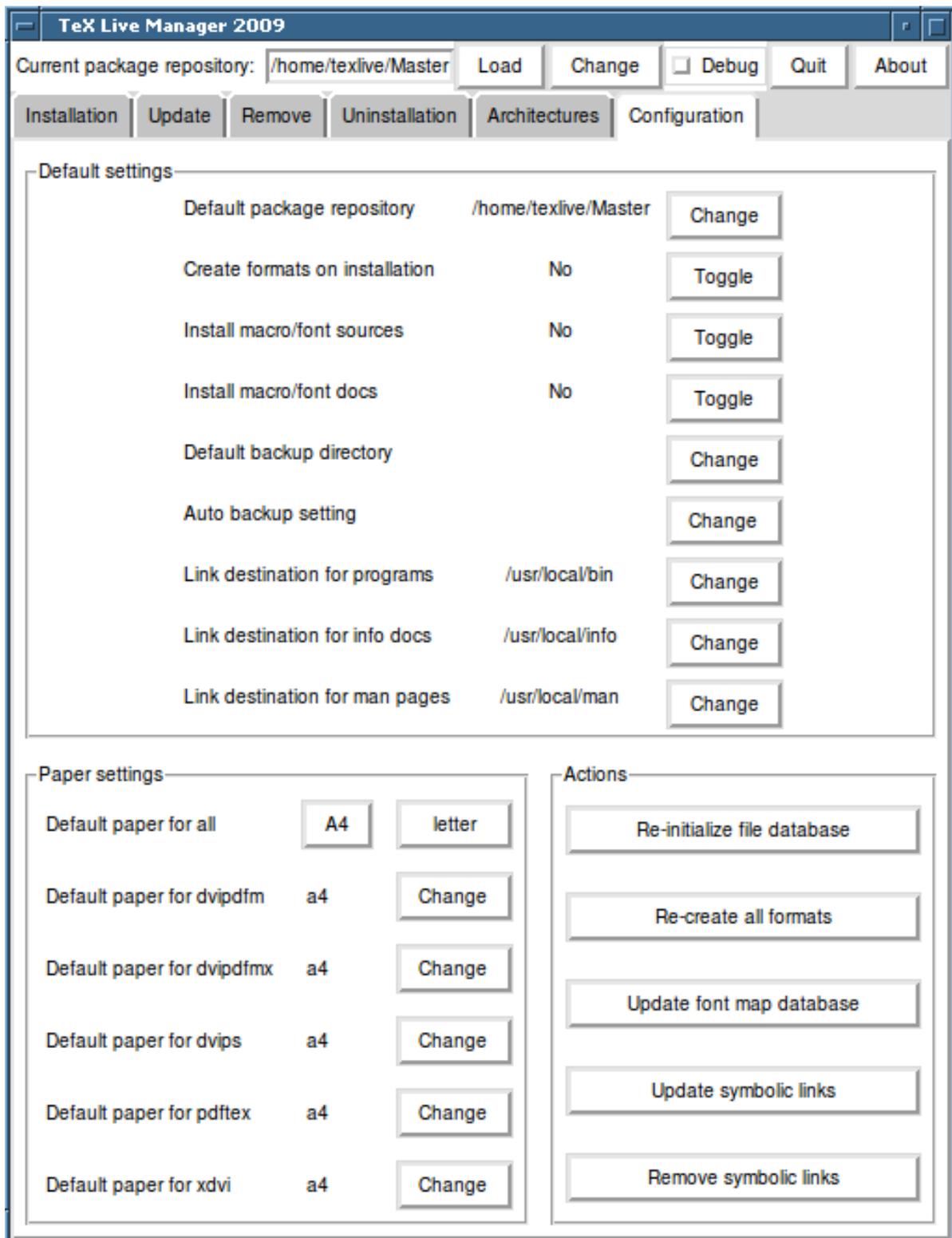


图 10: 图形界面模式的 tlmgr: Configuration 选项卡

6.2 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系统：

```
> 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):

```
> 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/2009/texmf/ls-R...
...
running fmtutil-sys --missing
...
Transcript written on xelatex.log.
fmtutil: /usr/local/texlive/2009/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安装后工作。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
```

会产生

```
package:    collection-latexextra
category:   Collection
shortdesc:  LaTeX supplementary packages
longdesc:   A large collection of add-on packages for LaTeX.
installed:  Yes
revision:   14675
```

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

```
> tlmgr -help
```

7 有关 Windows 平台的说明

TeX Live 拥有可运行于 Windows 和 Unix 平台的单一安装程序。当然, 只有在放弃对较老版本 Windows 的支持的情况下, 这才成为可能。因此, 现在 TeX Live 只能安装于 Windows 2000 或更新的版本。

7.1 针对 Windows 的特征

在 Windows 下, 安装程序额外地做了以下一些事情:

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、texdoctk、PS_View (psv), 它是 PostScript 预览程序) 和一些文档的菜单。PS_View 还有一个桌面快捷方式, 可以作为拖放 PostScript 文件的目标。

自动设置环境变量。 不再需要手动的配置步骤。

反安装程序。 安装程序为 TeX Live 创建了 ‘添加 / 删除程序’ 条目。这与 tlmgr 的删除按钮相对应。

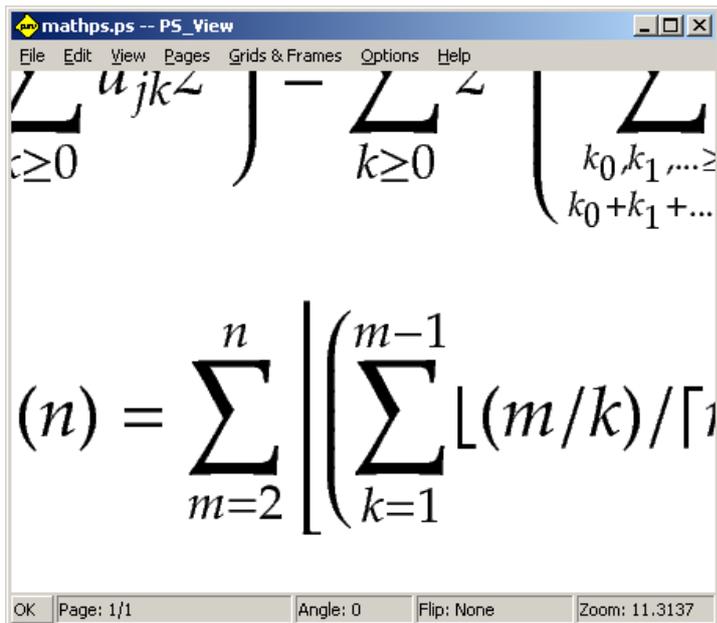


图 11: PS_View: 可以获得很高的放大倍数!

7.2 Windows 上附加的软件

为了使安装更加完整， $\text{T}_\text{E}\text{X}$ Live 需要支持那些 Windows 机器上不常见的软件包。 $\text{T}_\text{E}\text{X}$ Live 提供了以下缺失的部分：

Perl 和 Ghostscript。 由于 Perl 和 Ghostscript 的重要性， $\text{T}_\text{E}\text{X}$ Live 提供了这些程序的‘隐藏’拷贝。需要这些软件支持的 $\text{T}_\text{E}\text{X}$ 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，也不会与任何 Perl 或 Ghostscript 系统安装程序冲突。

PS_View。 同时安装的还有 PS_View，一款 PostScript 和 PDF 浏览器；见图 11。

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) 在线帮助中获得更加详细的信息。

TeXworks。 $\text{T}_\text{E}\text{X}$ works 是一个集成了 PDF 阅读器的 $\text{T}_\text{E}\text{X}$ 编辑器。它已经被配置好在 $\text{T}_\text{E}\text{X}$ Live 下工作。

命令行工具。 与常见的 $\text{T}_\text{E}\text{X}$ 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`、`unzip` 以及来自于 `xpdf` 套装中的一些命令行工具。虽然 `xpdf` 浏览器本身没有 Windows 版本，但你可以从 <http://blog.kowalczyk.info/software/sumatrapdf> 下载基于 `xpdf` 的 Sumatra PDF 浏览器。

fc-list、fc-cache 等。 来自于 `fontconfig` 库的这些工具有助于 $\text{XeT}_\text{E}\text{X}$ 处理 Windows 的系统字体。你可以使用 `fc-list` 来确定传递给经 $\text{XeT}_\text{E}\text{X}$ 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。

7.3 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 XP 和 Windows 2000 下，它通常位于 `C:\Documents and Settings\`；在 Windows Vista 下是 `C:\Users\`。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。

7.4 Windows 注册表

Windows 将几乎所有的配置数据存放在注册表中。注册表是包含几个主支的一系列分层组织的分支。对于安装的程序而言，最重要的主支是 `HKEY_CURRENT_USER` 和 `HKEY_LOCAL_MACHINE`，通常简写为 `HKCU` 和 `HKLM`。注册表的 `HKCU` 部分位于用户的主目录（参见 7.3 节）。`HKLM` 通常位于 Windows 目录的子目录中。

在某些情况下，系统信息可以从环境变量获得，但对于其它信息，如快捷方式的位置，则需要访问注册表。设置永久环境变量也需要访问注册表。

7.5 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整个操作系统。尽管如此，在实际中，作为受限用户与普通用户，你最好能够描述所处的用户类别：作为一个管理员是惯例，而不是特例。无论如何，我们努力使得 `TEX Live` 的安装不需要管理员权限。

如果用户是管理员身份，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，并修改系统环境变量。否则，安装程序只为当前用户创建快捷方式，并改变用户环境变量。

无论管理员状态如何，`TEX 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 `TEX` 已经存在搜索路径中，则可能会出现路径问题。因为有效路径是由系统路径后接用户路径组成，新安装的 `TEX 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 `TEX 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 `TEX Live`。如果安装了 `TEXworks`，其快捷方式也将 `TEX Live` 加进了搜索路径中，所以它应该不会出现问题。

对于 Vista 系统，还有另外一个奇怪的问题：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。

8 Web2C 用户指南

Web2C 是一整套 `TEX` 相关程序的集合：`TEX` 本身、`METAFONT`、`MetaPost`、`BibTEX`，等等。它是 `TEX Live` 的核心。Web2C 的主页及最新的手册等都在 <http://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 `TEX` 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、32 位 Windows 系统、Mac OS X 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 `TEX` 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 `TEX` 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。
 pktogf 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 T_EX。

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。
 --verbose 显示详细的执行过程。
 --version 显示版本信息，然后退出。

所有的 Web2C 程序均使用 Kpathsea (<http://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大量 T_EX 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 T_EX 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。

8.1 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 27 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。

要是没有这种数据库存在、又或者所有的数据库都不能和指定的路径前缀匹配上、又或者找到的数据库里没有进一步的匹配，就要搜索文件系统了（前提是我们没在路径前加上 '!!'，又或者搜索时就指定了这是一次“必定存在 (must exist)”型的搜索方式）。此时 Kpathsea 将根据路径元素构建一个需要检查的目录列表，逐个尝试这些目录，试图找到指定的文件。

搜索 '.vf' 文件和 T_EX 用 \openin 命令读入的文件时，会指定“文件必定存在 (must exist)”这个选项。而有些文件（比如 cmr10.vf）可能不存在，花费时间在磁盘上对它进行搜索是不值得的。因此，如果你安装了新的 '.vf' 文件后没有更新 ls-R，那这个文件将永远找不到。搜索时会优先在数据库寻找，然后再去搜索磁盘。一旦找到了就立即停止搜索，返回结果。

尽管最简单也最常见的路径元素是目录名称，Kpathsea 搜索的路径里还是可以使用其他额外功能的：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。所以我们将 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。

8.1.1 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 29 页的“调试操作”)。

8.1.2 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。而这个 `texmf.cnf` 存放的路径则是在 `TEXMFCNF` 变量里定义的，默认是在 `texmf/web2c` 目录下。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。所以，如果搜索路径是 `.$TEXMF`，那么文件 `./texmf.cnf` 里面的值要比 `$TEXMF/texmf.cnf` 里边的优先。

- 以 `%` 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[.programe] [=] value
```

‘=’ 号和空白都是可选的。

- `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 `'programe'` (程序名) 存在，则该定义只对正在运行的名叫 `programe` 或 `programe.exe` 的程序起作用。这可以让给不同种类的 `TEX` 程序设置不同的搜索路径。
- `value` (值) 允许任何 `%` 与 `@` 之外的字符出现。不支持在等号右侧使用 `$var.prog` 这样的写法。如果在 Unix 下，`value` 中的 `;` 字符会被转换为 `:`。如果你希望让 Unix, MS-DOS 和 Windows 里都用同一个 `texmf.cnf`，这会很有用。
- 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;}//
```

8.1.3 路径展开

和 Unix shell 类似，Kpathsea 能够识别搜索路径中的特殊字符。比如一个复杂的路径 `~$USER/{foo,bar}//baz`，将展开为这样的子目录：在 `$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。

8.1.4 默认展开

如果最高优先级 (参见第 25 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 ‘:’。

8.1.5 大括号展开

大括号展开是一项有用的特性, 其作用是把 `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 \TeX 层级结构。例如在 `texmf.cnf` 里有下面的定义 (这只是个近似的例子, 实际情况定义的目录树还要更多):

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

这样一来, 当你设置

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

的时候, 检查完当前目录后, 依次检查的路径是 `$TEXMFHOME/tex`, `$TEXMFLOCAL/tex`, `$TEXMFVAR/tex` 和 `$TEXMFMAIN/tex` (后两个只在 `ls-R` 数据库中搜索)。这样维护两套并行的 \TeX 结构就很方便的, 一套是“固定 (frozen)”的 (比如放在 CD 上) 而另一套是在新版本出现时就更新的。因为所有的定义里都用到了 `$TEXMF`, 所以你可以确信时常更新的那个版本肯定是先被找到的。

8.1.6 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。

8.1.7 特殊字符与其意义: 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:

- : 路径分隔符, 在路径的前边或者末尾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。
- {...} Brace expansion.
- // 子目录展开 (可以出现在除路径开头外的任意位置)。
- % 注释的起始。
- \ 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。

8.2 文件名数据库

Kpathsea 使用了一些方法来减少搜索时的磁盘访问次数。尽管如此，如果安装的文件足够多，在各个可能的目录下搜索某个文件仍然可能花上很长时间（尤其是在必须遍历数百个字体目录的时候）。因此，Kpathsea 使用一个专门构建的纯文本“数据库”文件，这个文件叫做 `ls-R`，它将文件和目录进行映射，避免对磁盘的大量搜索。

第二个数据库 `aliases` 允许你给 `ls-R` 中的文件指定其他的名字。有助于帮助源文件符合 DOS 8.3 命名规范。

8.2.1 文件名数据库

前边已经解释过，主文件名数据库的名称必须是 `ls-R`。你可以在每个需要搜索的 \TeX 目录树（缺省是 $\$TEXMF$ ）下放置一个这样的文件。Kpathsea 在 `TEXMFDBS` 指定的路径中寻找这些 `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。各类‘`mktex`’脚本也可能会间接调用这个脚本。实际上，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` 命令的输出格式必须正确（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。

如果文件在数据库中找不到，缺省情况下 Kpathsea 会在磁盘上搜索。如果某个特定的路径元素是以‘`!!`’起始的，就只会在针对这一元素的数据库中查找而不搜索磁盘。

8.2.2 kpsewhich: 独立的路径搜索

`kpsewhich` 程序将路径搜索从其他专用程序中独立出来，它可以作为类似 `find` 一样的程序，专门在 \TeX 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...
```

`option` 处的选项可以用‘`-`’也可以用‘`--`’来起始，并接受任何不造成疑义的缩写。

Kpathsea 将第一个非选项的参数作为文件名来查找，并返回找到的第一个文件。它不提供寻找所有相同名称文件的功能（你可以使用 Unix 的‘`find`’程序来达到这个功能）。

下面介绍了一些比较常见的选项。

- `--dpi=num` 将解析度设置为 `num`，这个选项只影响‘`gf`’文件和‘`pk`’文件的查找。为了与 `dvips` 兼容，提供‘`-D`’这个同义的参数，默认值是 600。
- `--format=name`
将查找的文件格式设置为 `name`。默认情况下是通过文件名来猜测格式的。对于扩展名有歧义性的格式，比如 `MetaPost` 支持文件和 `dvips` 配置文件，必须以 Kpathsea 已知的名称指定格式，比如 `tex` 或 `enc files`。运行 `kpsewhich --help` 会显示格式的列表。
- `--mode=string`
设置模式为 `string`，只影响‘`gf`’和‘`pk`’文件的查找。默认情况匹配所有的模式。
- `--must-exist`
尽一切可能找到文件，包括直接在磁盘上搜寻。默认情况下为了效率考虑，只检查 `ls-R` 数据库里的内容。
- `--path=string`
不通过文件名来猜测路径，沿 `string` 指出的路径搜索（也使用冒号分隔）。支持‘`//`’和所有常见的展开方式。‘`--path`’选项和‘`--format`’选项是互斥的。
- `--progname=name`
将执行查找的程序名称设为 `name`。这会通过 `.progname` 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。
```

8.2.3 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 (`TEX` 源文件目录下查找)。我们在 ‘`texmf-dist`’ 的 `tex/latex/base` 子目录下找到了这个文件。与之类似, 下列所有文件都顺利找到, 因为其扩展名没有二义。

```
> 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
```

另外, 最后一个是 `TUGBoat` 文章的 `BIBTEX` 参考文献数据库。

```
> kpsewhich cmr10.pk
```

.pk 类型的字体位图文件是给 `dvips` 或者 `xdvi` 这种显示程序提供的。因为 `TEX Live` 里没有预生成的 `Computer Modern` ‘.pk’ 字体, 所以没有找到任何内容 — `TEX Live` 默认使用 `Type 1` 版本的。

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

而这个字体 (`Washington` 大学的一套注音字母表) 就需要生成 ‘.pk’ 文件了。我们默认的 `META-FONT` 模式是 `ljfour`, 基础解析度是 `600 dpi (dots per inch)`, 所以会得到这样一个文件。

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

一旦指明我们需要寻找的只是 `300 dpi` 的文件时 (`-dpi=300`), 就会发现系统里没有符合要求的文件。这样 `dvips` 或 `xdvi` 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, `tex.pro` prologue 文件, 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’ 后缀可能会有二义, 我们必须指明意思是 `dvips config`。

```
> 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
```

这样我们可以更仔细地分析一下 `URW Times` 的 `PostScript` 支持文件。依照标准的字体命名方案, 这些文件的前缀是 ‘utm’。首先查询的是配置文件, 其中包含的是 `map` 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 集合中的 Type 1 PostScript 字体文件名。内容差不多是下边这样 (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
utmbo8r NimbusRomNo9L-Medi    ... <utmb8a.pfb
utmro8r NimbusRomNo9L-Regu    ... <utmr8a.pfb
```

比如我们可以以一个 Times Roman 字体 `utmr8a.pfb` 为例，在 `texmf` 目录下的 Type 1 字体文件中寻找它的位置:

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

现在你可以发现，追寻某个文件的下落是如此方便。尤其是在你怀疑找到的文件错了的时候，这些功能非常重要，因为 `kpsewhich` 只能告诉你找到的第一个文件。

8.2.4 调试操作

有时你可能会需要分析程序是如何解析文件引用的。`Kpathsea` 提供了多层调试输出来实现这个功能:

- 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。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。

等级 `-1` 将启用上述所有选项，实际上这是最方便的设置方法了。

类似地，在 `dvips` 程序中设置上述调试选项的组合，你就可以出 `dvips` 是从哪里找到它所需的文件的。另一方面，如果找不到某个文件，调试输出也会显示程序从哪些目录进行了查找，你也可以据此判断出问题出在哪里。

一般而言，因为所有的程序都在其内部调用 `Kpathsea` 库，你可以设置 `KPATHSEA_DEBUG` 环境变量来选择调试参数，将这个变量设置为上述参数的组合 (加法) 以获得对应的功能。

(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`)。

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

```

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/.dviplib, must_exist=0, find_all=0,
  path=./tex:!!/usr/local/texmf/dvips//).
search(file=/home/goossens/.dviplib, must_exist=1, find_all=0,
  path=./tex/dvips://!!/usr/local/texmf/dvips//).
kdebug:search($HOME/.dviplib) =>
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

```

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

```

图 13: 寻找 prolog 文件

```

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]

```

图 14: 寻找字体文件

```

\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
```

此时我们将 `dvips` 的调试等级 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，texmf/doc/html/dvips/dvips_toc.html）。（稍作整理的）输出见图 12。

`dvips` 启动后就开始搜寻其需要使用的文件。首先找到的是 `texmf.cnf`，它给出了用于进一步查询其他文件的路径，然后找到的是 `ls-R` 文件名数据库（用于优化文件搜索速度）和 `aliases` 文件（用于创建同一文件的多个别名，比如为较自然的长文件名创建 DOS 8.3 风格的短别名）。然后 `dvips` 去寻找它的通用配置文件 `config.ps`，接下来查找个人定制文件 `.dviplib`（不过上述例子中并未找到，显示 *not found*）。最后 `dvips` 找到了 Computer Modern PostScript 字体的配置文件 `config.cms`（因为执行

dvips 时使用了 `-Pcms` 选项)。这个文件里包含了字体的 \TeX 名称, PostScript 名称和文件名的对应关系。

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

于是 dvips 接下来寻找这些文件, 再加上固定载入的, 通用映射文件 `psfonts.map` (这个文件里包含常用 PostScript 字体的映射, 参见第 8.2.3 节的最后一部分, 讨论了 PostScript 映射文件的处理)。

这时候 dvips 向用户表示它的存在:

```
This is dvips(k) 5.92b Copyright 2002 Radical Eye Software (www.radicaledge.com)
```

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
```

找到所需文件后, dvips 输出日期和时间, 并告知我们它将生成 `hello-world.ps` 文件, 需要用到 `cmr10` 字体文件, 这个字体属于“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.
```

寻找并找到了 `cmr10.tfm` 文件。然后再次引用了一些 prolog 文件 (此处略去), 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]
```

8.3 运行时选项

Web2C 另一项有用的特性是可以通过 `Kpathsea` 读入的运行时文件 `texmf.cnf` 来控制一系列的内存参数 (具体而言是数组的大小)。内存的设置可以在这个文件的第三部分找到。比较重要的几个设置是:

`main_memory` 总的可用内存数, 以 word 为单位, \TeX , METAFONT 和 MetaPost 受此限制。每修改一次都必须重新生成一个新的格式文件。比如你可以生成一个“巨型”版本的 \TeX , 将其存为 `hugetex.fmt`。

`extra_mem_bot` 为“大型” \TeX 数据结构预留的额外空间: `boxes`, `glue`, `breakpoint` 等。如果你使用 $\Pi\TeX$ 时特别有用。

`font_mem_size` \TeX 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。一般散列表足够存储将近 10,000 个控制序列。如果你排版的是一本有大量交叉引用的书籍, 这个值可能不够用。默认的 `hash_extra` 是 50000。

当然，这项功能并非真正的动态内存和数组分配的替代，但考虑到动态分配在现在的 $\text{T}_{\text{E}}\text{X}$ 太难实现，才通过这些选项提供了一些灵活性。

9 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 <http://tug.org/usergroups.html>。)
- CTAN 团队, 值得一提的是 Robin Fairbairns, Jim Hefferon, 和 Rainer Schöpf。他们负责分发 $\text{T}_{\text{E}}\text{X}$ Live 光盘镜像, 为软件包更新提供支撑, $\text{T}_{\text{E}}\text{X}$ Live 正是基于这些软件包构建的。
- 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。
- 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。
- Jin-Hwan Cho 和整个 DVIPDFMx 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 $\text{t}_{\text{E}}\text{X}$ 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。Eitan 于 2009 年 6 月过早地离开了我们, 我们希望能把这份文档献给它, 以志纪念。
- Hans Hagen, 他对 Con $\text{T}_{\text{E}}\text{X}$ t 格式 (<http://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。
- Hàn Thê Thành, Martin Schröder, 和 pdf $\text{T}_{\text{E}}\text{X}$ 团队, 他们持续不断地改进 $\text{T}_{\text{E}}\text{X}$ 的功能。
- Hartmut Henkel, 他对 pdf $\text{T}_{\text{E}}\text{X}$ Lua $\text{T}_{\text{E}}\text{X}$ 等程序的开发起到了重要的贡献。
- Taco Hoekwater, 他对 MetaPost 和 (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 Con $\text{T}_{\text{E}}\text{X}$ t 融入 $\text{T}_{\text{E}}\text{X}$ Live、给 Kpathsea 添加多线程功能, 等等许多工作。
- Paweł Jackowski, 他创建了 Windows 下的安装程序 tlpm, 和 Tomasz Luczak, 因为他的 tlpm gui 被用在了以前的版本。
- Akira Kakuto, 以他的 W32 $\text{T}_{\text{E}}\text{X}$ 发行版 (<http://w32tex.org/>) 为基础提供了 Windows 下的二进制文件, 以及许多其他的贡献。
- Jonathan Kew, 他开发了非凡的 Xe $\text{T}_{\text{E}}\text{X}$ 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 Mac $\text{T}_{\text{E}}\text{X}$ 安装程序的早期版本, 还有我们推荐的前端 $\text{T}_{\text{E}}\text{X}$ works。
- Dick Koch 他维护了 Mac $\text{T}_{\text{E}}\text{X}$ (<http://tug.org/mactex>) 这个和 $\text{T}_{\text{E}}\text{X}$ Live 联系非常紧密的发行版本。
- 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。
- 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。
- 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, 他创建了 $\text{T}_{\text{E}}\text{X}$ Live 最早的 Windows 支持。
- Norbert Preining, 他是 $\text{T}_{\text{E}}\text{X}$ Live 2008 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), 并提供了许多建议。
- Sebastian Rahtz, 他是 $\text{T}_{\text{E}}\text{X}$ Live 的创始者, 并维护了它许多年。
- Phil Taylor, 他设置了 BitTorrent 下载。
- Tomasz Trzeciak, 他为 Windows 开发提供了广泛的帮助。
- Vladimir Volovich, 他极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 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。
- Olaf Weber, 他对 Web2C 耐心的维护。
- Gerben Wierda, 他创建和维护了原来的 Mac OS X 支持, 并参与了大量集成和测试工作。

- Graham Williams, 他整理了软件包的依赖情况的 $\text{T}_{\text{E}}\text{X}$ Catalogue。

二进制版本的编译器: Peter Breitenlohner (x86_64-linux), Karl Berry (i386-linux, sparc-linux), Ken Brown (i386-cygwin), Akira Kakuto (win32), Dick Koch (universal-darwin), Norbert Preining (alpha-linux), Jukka Salmi (i386-netbsd), Thomas Schmitz (powerpc-linux), Apostolos Syropoulos (i386-solaris), Vladimir Volovich (powerpc-aix, sparc-solaris), Olaf Weber (mips-irix)。关于 $\text{T}_{\text{E}}\text{X}$ Live 编译进程的信息, 请查阅 <http://tug.org/texlive/build.html>。

当前的文档和翻译更新: Jjgod Jiang, Jinsong Zhao, Yue Wang, & Helin Gai (中文), Klaus Höppner (德语), Manuel Pégourié-Gonnard (法语), Petr Sojka & Jan Busa (捷克/斯洛伐克语), Boris Veytsman (俄语), Staszek Wawrykiewicz (波兰语)。 $\text{T}_{\text{E}}\text{X}$ Live 文档的主页是 <http://tug.org/texlive/doc.html>。

当然, 最重要的感谢应该致予 Donald Knuth, 感谢他发明了 $\text{T}_{\text{E}}\text{X}$, 也感谢他将 $\text{T}_{\text{E}}\text{X}$ 赠与全世界。

10 发行历史

10.1 过去

1993 年末荷兰 $\text{T}_{\text{E}}\text{X}$ 用户组开始为 MS-DOS 用户开发 4All $\text{T}_{\text{E}}\text{X}$ 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 4All $\text{T}_{\text{E}}\text{X}$ CD, 同时 TUG 技术委员会工作组也开始设计 $\text{T}_{\text{E}}\text{X}$ 目录结构 (<http://tug.org/tds>), 以指明如何创建一套一致而可控的集合, 囊括所有 $\text{T}_{\text{E}}\text{X}$ 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。4All $\text{T}_{\text{E}}\text{X}$ 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 $\text{T}_{\text{E}}\text{X}$ 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 $\text{teT}_{\text{E}}\text{X}$ 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 $\text{teT}_{\text{E}}\text{X}$ 的特性囊括在内, 这样我们决定在 $\text{teT}_{\text{E}}\text{X}$ 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, $\text{teT}_{\text{E}}\text{X}$ 的一个新版本出现了, $\text{T}_{\text{E}}\text{X}$ 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 $\text{teT}_{\text{E}}\text{X}$ 和新版本的 Web2C (7.3)。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 $\text{T}_{\text{E}}\text{X}$ Live 5 的首要变化还是移除了所有的非自由软件。 $\text{T}_{\text{E}}\text{X}$ Live 的所有部分现在都在向 Debian Free Software Guidelines 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 Mac OS X 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 $\text{teT}_{\text{E}}\text{X}$ 集成, 因为在第 5 和第 6 版中它们偏离得太远了。

10.1.1 2003

2003 年, 在更新和增添持续不断到来的情况下, 我们发现 $\text{T}_{\text{E}}\text{X}$ 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.4)。此外:

- 在 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 团队的要求下, 我们将 `latex` 和 `pdflatex` 命令改为使用 ϵ - $\text{T}_{\text{E}}\text{X}$ 引擎 (参见 p.6)。
- 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 HPUX 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 T_EX 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。
- T_EX, 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 T_EX 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 T_EX 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: T_EX Live 2003。

10.1.2 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 <http://tug.org/texlive/mapenc.html>。
- 因为有人可能更愿意使用 MiK_TE_X 而非 Web2C 系统, T_EX Collection 现在包含了一套基于 MiK_TE_X 的可安装 CD, 参见第 2 节 (p.4)。
- 在原来旧版本 T_EX 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.5) 及各目录下的 README 文件。
- 所有 T_EX 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, 并且可以通过 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 T_EX (只将位置 32-126 的字符视为可见), ConT_EXt (0-255 都视为可见) 和与 Ω 相关的格式。缺省的情况几乎与 T_EX 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/doc/web2c/web2c.html#TCX-files (另外, 如果遇到 Unicode 输入, T_EX 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。)
- pdfetex 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 L^AT_EX, ConT_EXt 等格式中使用, 另外 ε-T_EX 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 L^AT_EX 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。
- pdfT_EX (<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的选项现在都必须改用 T_EX 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 pdfT_EX 手册以了解更多信息: texmf/doc/pdftex/manual。

- `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/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/](http://www.olsak.net/encTeX.html) 和 <http://www.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](http://www.tex.ac.uk/cgi-bin/texfaq2html?label=aleph) 和 <http://www.tex.ac.uk/cgi-bin/texfaq2html?label=aleph> 找到。Aleph 的 `LATeX` 格式文件称做 `lamed`。
- 最新发布 `LATeX` 包含了是新版 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 texmf/doc/man/man1/dvipng.1。
- 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 (<http://www.ctan.org/tex-archive/fonts/greek/cb>)。
- 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-opensbsd` 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。

10.1.3 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, 用于修改系统目录树下的配置。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 5 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 `LATeX` 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 `gsview`，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。

10.1.4 2006–2007

2006–2007 年， \TeX Live 的一个重大变化是增加了 Xe \TeX ，以 `xetex` 和 `xelatex` 程序的形式提供。请参见 <http://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 (<http://tug.org/metapost/articles>)，pdf \TeX 同样如此 (<http://tug.org/applications/pdftex>)。

\TeX `.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。（ \LaTeX 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 (<http://www.ctan.org>)。

从内部角度看，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 (<http://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。

10.1.5 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.tlpsdb` 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 6 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 <http://tug.org/TUGboat/Articles/tb29-2/tb92knut.pdf>。

10.2 现状

在 2009 年，为了充分利用 Lua \TeX 的 OpenType 支持等特性，Lua(LA) \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 <http://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 <http://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，<http://tug.org/texworks>。设计这个跨平台前端的灵感来自于 Mac OS X 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 <http://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。`DVIPDFMx` 项目的主页在 <http://project.ktug.or.kr/dvipdfmx>。

现在包括了 cygwin 和 i386-netbsd 平台下的可执行文件，但其他 BSD 发行版的可执行文件被去掉了；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` (<http://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。

10.3 未来

TeX Live 并不完美！（也永远不会达到完美。）我们希望继续发行新的版本，也希望提供更多的帮助文档、更多的实用程序、更多的安装程序，当然还有更多更新的宏包与字体。这个工作是由压力巨大的志愿者在其空闲时间完成的，也有很多不够完善的地方。请参见 <http://tug.org/texlive/contribute.html>。

请把更正、建议或者提供帮助的意愿发送到：

`tex-live@tug.org`
<http://tug.org/texlive>

祝你使用 \TeX 愉快！

11 翻译说明

这里对简体中文版本《 \TeX 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 \TeX 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 $\text{T}_{\text{E}}\text{X}$ 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。