TEX Live 指南
TEX Collection 2006

Karl Berry 编写

http://tug.org/texlive/

2006 年 12 月

目录

表格列表

1 简介

本文档描述 TeX Live 软件的主要功能和特性，TeX Live 是 TeX 及其相关程序在 GNU/Linux 及其他类 Unix 系统、Mac OS X 和 (32 位) Windows 系统下的一套发行版。(注意：在旧的 Mac OS 或 MS-DOS 系统下，TeX Live 恐怕不太有用。)

TeX Live 包括了 TeX, LaTeX 2$_\varepsilon$, ConTeXt, METAFONT, MetaPost, BibTeX 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。它是 TEX Collection 这套比它更为庞大的集合的一部分 (在第 4 页的第 2 节对 TEX Collection 有简短的介绍)。这两套发行版都是由 TEX 用户组织协作完成的。

如果你希望找到这里提供的软件包的更新版本，请查阅 CTAN: `http://www.ctan.org`。

文档末尾的第 10 节 (第 37 页) 介绍了这一版 TEX Live 的重要改变。

1.1 TEX Live 的基本使用

你可以通过下列三种方式来使用 TEX Live:

1. 直接从 live DVD 上运行 TEX Live (参阅第 2.1 节，第 4 页)。这种情况几乎不花费任何磁盘空间，就能让 TEX Live 的所有内容立即供你支配。当然，执行的效率会比从本地磁盘上运行要低一些，不过这种方式仍然是很有用的。
2. 从 DVD 或者 inst CD 上把 TEX Live 部分或完整地安装到本地磁盘。这是 TEX Live 最常见的使用方式了。最小的安装大概需要 100 MB，完整安装则需要将近 1.3 GB。
3. 将 TEX Live 中的某个或某些软件包单独安装到你现有的 TEX 系统中，不管这个系统是以前安装的 TEX Live 还是别的什么发行版。

上述内容都在针对各个操作系统的安装章节里有详细的介绍，不过下面的导引也许能帮助你以最快的速度使用起来:

- Unix 和 Mac OS X 系统使用的主安装脚本是 `install-tl.sh`。
  GNU/Linux 用户还能尝试一套新的 GUI 模式安装程序: 执行 `tlpmgui`。关于它的信息可以在第 16 页的第 6 节找到。
- 安装独立软件包使用的脚本是 `install-pkg.sh`。
- Windows 下使用的安装程序是 `tlpmgui.exe`。这个程序也可以用于添加或删除软件包。参考下面的第 6 节以了解更多信息。

1.2 获得帮助

TEX 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 TEX Live，改为购买商业 TEX 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 TEX，`http://tug.org/begin.html` 这个网页提供了这个系统的简短介绍。

TEX FAQ 这套庞大的 TEX 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，它包含在 TEX Live 的 `texmf-doc/doc/english/FAQ-en` 目录下，也可以在 `http://www.tex.ac.uk/faq` 网站上找到它。有问题时请先看看这里能否找到解答。

TEX Catalogue 如果你在寻找某个特定的宏包、字体、程序等等，TEX Catalogue 就是该找的地方。这里是所有 TEX 相关内容的一个巨大集合。参见 `texmf-doc/doc/english/catalogue`，或者 `http://www.ctan.org/tex-archive/help/Catalogue`.

TEX 网上资源 `http://tug.org/interest.html` 页面上有许多和 TEX 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。

TEX Live 技术支持 如果你需要报告 bug，或者提出对 TEX Live 的发行、安装或文档的建议和意见，可以是用 `tex-live@tug.org` 这个邮件列表。不过，如果问题是针对 TEX 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。

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。欢迎来到 TEX 世界!

2 TEX Live 的结构

这个小节描述的是 TEX Live 的结构与内容，以及包含 TEX Live 的 TEX Collection。

2.1 多种发行版：live, inst, protext

CD-ROM 格式的空间所限，我们不得不将 TEX Collection 切分为如下几个发行版:

live 是在 DVD 上的完整系统；因为内容过多，无法放在 CD 上。这个版本可以直接运行，也可以安装到硬盘中。它还包括 CTAN 仓库的一个快照 (snapshot)、Windows 下使用的 protext 发行版和 Mac OS X 下使用的 MacTEX 发行版，这些内容都和 TEX Live 是相互独立的，同时，它还将其他各种软件包归在 `texmf-extra` 目录下。

CTAN, protext, MacTEX, 和 `texmf-extra` 和 TEX Live 的版权规则是不同的，所以在分发或修改这些软件时请小心。

inst(allable) 是一套在 CD 上的可安装的完整系统。为了让它容纳在 CD 中，程序和宏包都被压缩存储。因此，你必须把它安装在硬盘上才能运行 TEX，无法直接从 CD 执行 (由此得名)。安装步骤在后续章节有介绍。

protext 是 Windows 下的 MiKTEX 发行版的一个增强版本。ProTEXt 在 MiKTEX 基础上增加了一些额外的工具，简化了安装过程。它是完全独立于 TEX Live 的，有其自己的安装步骤，可以在光盘上直接执行，也可以安装到硬盘上。ProTEXt 的主页在 `http://tug.org/protext`。

ProTEXt 同时在 live DVD 的根目录下和其单独的 CD 中提供 (考虑到有人可能无法使用 DVD)。

要分辨你正在使用的是哪种发行版，你可以查阅 TEX Live 目录下的 `00`*type*`.TL` 文件 (通常排在文件列表的最前面)。这个文件中还包含有 TEX Live 的发布日期。

诚然，每个 TEX 用户组织都可以根据自己的意愿选择究竟发行什么版本。(TUG 会给其成员寄去全部三张光盘。)

2.2 顶层目录

这里是对 TEX Live 发行版顶层目录的一个简短的列表和描述。注意在 live DVD 中，整个 TEX Live 目录结构都存放于 texliveYYYY (YYYY 表示年份) 子目录中，而非光盘的顶层目录下。

bin	TEX 系统程序，按平台组织。
source	所有程序的源代码，包括主要的 Web2C TEX 和 METAFONT 发行版。这些内容存放在一个 bzip2 格式归档压缩包中。
setuptl	Linux 和 Windows 下使用的安装程序。
support	各类辅助性宏包与程序。这些内容**不会**自动安装。包括了各类编辑器和 TEX 的前端 (外壳) 程序。
texmf	存放程序、附属文件和文档的目录树。不包括 TEX 格式文件和宏包。 (在下一小节中称为 TEXMFMAIN。)
texmf-dist	存放格式文件 (format file) 和宏包的主目录树。 (在下一小节中称为 TEXMFDIST。)
texmf-doc	这个目录树用于存放和具体程序、宏包关系不大的纯粹文档，按语言组织。
texmf-var	用于存放自动生成的文件的目录树。 (在下一小节中称为 TEXMFSYSVAR。)

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。

尽管 texmf-doc 目录只包含文档，但并非全部的文档都在这个目录下。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。与之类似，TEX 宏包与格式文件的文档放在 texmf-dist/doc 目录。但不管放在哪个地方，你都可以使用 texdoc 或 texdoctk 程序来寻找这些文档。顶层目录下的 doc.html 文件中提供的完整的链接也会有所帮助。

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。texconfig 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，包括辅助脚本 (比如 web2c/mktexdir)、pool 文件和其他辅助性文件。

TEXMFDIST 存储主要的宏包、字体等等，即一开始就在发行版中包括的而非自己后续添加的那些文件。

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的宏包、字体等等。这个变量的展开默认是由 $HOME 决定的，即根据不同的用户选择不同的主目录。

TEXMFCONFIG 给 texconfig、updmap、和 fmtutil 这些 teTEX 实用程序存储修改过的配置文件。默认在 $HOME 目录下。

TEXMFSYSCONFIG 给 texconfig-sys、updmap-sys、和 fmtutil-sys 等 teTEX 实用程序存储存储修改过的配置文件，这些配置是对全系统都有效的，而不止当前用户。

TEXMFVAR 这个目录是给 texconfig、updmap 和 fmtutil 存储 (缓存) 格式文件、生成的 map 文件这类运行时数据的。默认是在 $HOME 目录下。

TEXMFSYSVAR 给 texconfig-sys、updmap-sys 和 fmtutil-sys 这三个命令存储、缓存运行时使用的格式文件和生成的 map 文件。

在第 13 页的第 4.1 节有对 texconfig 等相关实用程序的更多讨论。

2.4 TEX 的扩展版本

TEX Live 包括下列 TEX 扩展版本:

ε-TEX 为 TEX 增加了一套小而有用的新原语 (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--XET 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`。

pdfTEX 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。参见其文档 `texmf/doc/pdftex/manual/` 和示例文件 `texmf/doc/pdftex/manual/samplepdf/samplepdf.tex`。除 plain TEX 外，所有的格式文件都默认使用 pdfTEX 来执行。

XeTEX 通过第三方库，增加对 Unicode 输入文本和 OpenType 字体的支持。参见 `http://scripts.sil.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。参见 `texmf-dist/doc/omega/base/doc-1.8.tex` (文档的更新可能不是很及时)。

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 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。

dv2dt, dt2dv DVI 和纯文本之间的相互转换。

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。

lacheck LATEX 语法检查器。

texexec ConTEXt 和 PDF 处理工具。

tex4ht TEX 到 HTML 的转换器。

3 Unix 系统下的安装

正如第 1.1 节 (p. 3) 所提到的，TEX Live 有这三种用法:

1. 直接从光盘执行。
2. 安装到硬盘。

3. 将某个或某些软件包安装到现有的 TEX 安装中。

以下章节就是针对 Unix 系统介绍每一种用法的详细步骤。

警告： TEX Collection CD 与 DVD 使用 ISO 9660 (High Sierra) 格式，**加上** Rock Ridge (Windows 下是 Joliet) 扩展。因此，在 Unix 下为了使用 TEX Collection 的全部功能，你的系统必须支持 Rock Ridge 扩展。请查阅你的 mount 命令的文档以了解如何实现。如果你是在一个有多台机器的局域网中使用，可以在支持 Rock Ridge 的那台机器上挂载光盘，这样同样可以用于其他所有机器。

现代的操作系统应该都可以毫无问题的使用这些光盘，如果的确有什么问题，请告知我们。下面的讨论都是基于你挂载的 CD 可以完全兼容 Rock Ridge 的情况下进行的。

3.1 从光盘直接运行 TEX Live (Unix 系统下)

不安装到磁盘而直接在 live DVD 上使用 TEX 系统是可行的 (这正是 TEX `Live' 这个名字的来源)。但直接从其他 CD 运行 TEX 是**不**可能的 (参见第 2.1 节，p. 4)。 以启用 Rock Ridge 扩展的方式挂载好 CD 或 DVD 之后，就可以开始使用了。具体的命令可能依系统的不同而有所不同，下面的例子在 Linux 下可以正常工作，不过设备的名称 (此处是 /dev/cdrom) 可能会有区别。(所有我们的例子都使用 > 作为 shell 提示符；所有用户的输入都有 下划线。)

```
> mount -t iso9660 /dev/cdrom /mnt/cdrom
```

将当前目录更改为挂载点：

```
> cd /mnt/cdrom
```

Mac OS X 系统通常在 /Volumes 目录下列出自动挂载好的光盘。 执行安装脚本 install-tl.sh：

```
> sh install-tl.sh
Welcome to TeX Live...
```

在几行欢迎信息和列出主菜单选项之后，安装程序会要求你输入一个命令。此时输入对应的字符，按下回车即可，不必输入选项周围的尖括号。命令大小写均可，在示例中我们使用的是小写。

因为要直接从光盘上执行，我们的第一个命令是 d，然后用第二个命令 1 来设置路径。就算是在光盘直接执行，我们仍然需要设置一个在本地磁盘上的目录，用于存放 TEX 系统自动生成的文件，例如字体和格式文件，并存放修改过的配置文件，如果你需要修改的话。

在本例中，我们使用 /opt/texlive2006。在路径中包含年份是有好处的，毕竟生成的文件未必是和每年发行的 TEX Live 都兼容的。(如果你能接受默认值 /usr/local/texlive/2006，那也可以直接跳过这一步。)

```
Enter command: d
Current directories setup:
<1>  TEXDIR:     /usr/local/texlive/2006
...
Enter command: 1
New value for TEXDIR [/usr/local/texlive/TeX]: /opt/texlive2006
...
Enter command: r
```

返回主菜单后，我们输入第二个命令 r，这也是需要输入的最后一个命令。它用于设置直接从光盘执行而不安装到硬盘：

```
Enter command: r
Preparing destination directories...
...
Welcome to TeX Live!
>
```

如上所示，现在我们又回到了系统提示符下。

下面你需要设置两个环境变量：将 PATH 设置为一个平台相关的值 (这样我们才能找到要运行的程序)，将 TEXMFSYSVAR 设置为上面指定的路径值。参考表 1 以决定对应不同机器架构的 TEX Live 系统版本。

主要的安装过程完成，又设置好环境变量之后，最后一步就是执行 texconfig 或者 texconfig-sys 来定制你的安装了。在第 4.1 节 (p. 13) 中对此有详细解释。

表 1: 支持的系统架构

alpha-linux	HP Alpha GNU/Linux
hppa-hpux	HPPA HP-UX
i386-darwin	x86 Mac OS X
i386-freebsd	x86 FreeBSD
i386-linux	x86 GNU/Linux
i386-openbsd	x86 OpenBSD
i386-solaris	x86 Solaris
mips-irix	SGI IRIX
powerpc-aix	IBM RS/6000 AIX
powerpc-darwin	PowerPC Mac OS X
powerpc-linux	PowerPC GNU/Linux
sparc-linux	Sun Sparc GNU/Linux
sparc-solaris	Sun Sparc Solaris
win32	Windows (32-bit)
x86_64-linux	x86 64-bit GNU/Linux

环境变量的格式和用于永久设置环境变量的初始化文件是和你使用哪个 shell 有关的。如果你使用的是 Bourne 兼容型 shell (比如 sh、bash、ksh 之类)，就应该将下面的内容放在你的 $HOME/.profile 文件中：

```
PATH=/mnt/cdrom/bin/archname:$PATH; export PATH
TEXMFSYSVAR=/opt/texlive2006/texmf-var; export TEXMFSYSVAR
```

而对于 C shell 兼容型的 shell (csh、tcsh)，应该将下面的内容放在你的 $HOME/.cshrc 文件中：

```
setenv PATH /mnt/cdrom/bin/archname:$PATH
setenv TEXMFSYSVAR /opt/texlive2006/texmf-var
```

然后注销当前用户再重新登录，就可以测试你的安装是否成功了 (参见第 14 页的第 4.2 节)。

如果对上述步骤有何疑问，请咨询你那儿对系统配置熟悉的专家，看他能否帮你解决。比如如何挂载 TEX Live 光盘、安装到哪个或哪些目录、修改个人初始化文件的正确方法等等，都有可能依系统配置的不同而有区别。

3.2 将 TEX Live 安装到硬盘

将 TEX Live 发行版安装到硬盘上当然是可行的，这也是最典型的一种使用方式。live 和 inst 发行版都支持这种方式。(关于不同发行版本的介绍，参见第 4 页的第 2.1 节。)

以启用 Rock Ridge 扩展的方式挂载好 CD 或 DVD 之后，就可以开始使用了。具体的命令可能依系统的不同而有所不同，下面的例子在 Linux 下可以正常工作，不过设备的名称 (此处是 `/dev/cdrom`) 可能会有区别。(所有我们的例子都使用 > 作为 shell 提示符；所有用户的输入都有下划线。)

```
> mount -t iso9660 /dev/cdrom /mnt/cdrom
```

将当前目录更改为挂载点：

```
> cd /mnt/cdrom
```

Mac OS X 系统通常在 `/Volumes` 目录下列出自动挂载好的光盘。 执行安装脚本 `install-tl.sh`：

```
> sh install-tl.sh
Welcome to TeX Live...
```

在几行欢迎信息和列出主菜单选项之后，安装程序会要求你输入一个命令。此时输入对应的字符，按下回车即可，不必输入选项周围的尖括号。命令大小写均可，在示例中我们使用的是小写。

表 2 简略地列出了主菜单提供的所有选项。执行这些选项的顺序可以任意，但 `i` 必须放在最后。不妨按其列出的顺序来执行。

表 2: 安装的主要选项

`p`	你正在运行的系统平台 (机器架构)。
`b`	需要安装的程序 (二进制文件) 对应机器架构。
`s`	基本的安装方案 (最小安装，推荐安装，完整安装，等等)。
`c`	在基本方案的基础上对个别软件集合的安装进行修改。
`l`	修改需要支持的语言种类。
`d`	安装到的路径。
`o`	其他选项。
`i`	进行安装。

下面是每个选项的进一步细节。

`p` -- 当前平台 因为安装脚本可以自动探测其执行的平台，通常这个选项都不必设置。你可以用这个选项可以更正自动探测的结果。

`b` -- 二进制文件的架构 默认情况下，只会安装对应你当前平台的那些二进制文件。你可以在这个菜单中选择安装其他平台下的二进制文件 (或者取消安装当前平台的)。这对那些在一个网络中多台不同架构机器共享一个 TEX 环境的情况特别有用。请参考第 8 页的表 1 以了解支持的所有架构。

`s` -- 基本安装方案 在这个菜单里你可以通过选择一套安装方案，来对软件包集合作整体的选择，默认的方案是 `full`，即安装所有东西。不过可以选择 `basic` 方案来只安装一个最小的系统，或者是 `medium` 这个折中方案。这里还有专门针对 Omega 和 XML 的集合选择。

c -- 个别软件包集合 这个菜单里你可以覆盖上述方案中选择安装的软件包集合。``集合''是一个比``方案''更细一层的概念---集合包含一个或多个软件包，而软件包 (TEX Live 里最底层的分组) 包含的是具体的 TEX 宏文件、字体族等等。这个菜单里用于选择的字母是大小写敏感的。

l -- 语言集合 这个菜单和 c 的基本目标是一致的，也是覆盖已选方案的默认设置。只不过这里是专为选择语言而设。用于选择的字母命令也是大小写敏感的。这里是 TEX Live 支持的语言集合：[1]

(some) African scripts	Arabic	Armenian	Chinese Japanese Korean
Croatian	Cyrillic	Czech/Slovak	Danish
Dutch	Finnish	French	German
Greek	Hebrew	Hungarian	Indic
Italian	Latin	Manju	Mongolian
Norwegian	Polish	Portuguese	Spanish
Swedish	Tibetan	UK English	Vietnamese

语言集合通常包括字体、宏、断字规则 (hyphenation pattern) 和其他辅助性文件。(比如一旦你选择了 French 集合，就会安装 frenchle.sty。) 同时，安装的语言集合还会影响 language.dat 配置文件，这个文件用于控制载入的断字规则。

d -- 安装路径 这里有三个路径可以修改：

TEXDIR 顶层目录，其他所有内容都将被安装到这个目录下。默认值是 /usr/local/texlive/2006，通常会有改动。我们建议在这个目录名中包含年份，以区分不同版本的 TEX Live。(不过也许你可以创建类似 /usr/local/texlive 这样的符号链接，这样一旦新版本发布了，你也可以方便的更新。)

在 Mac OS X 下，大多数的 TEX 前端软件都在 /usr/local/teTeX 目录下寻找 TEX 程序，所以你可能更倾向于把 TEX Live 安装到那里。

TEXMFLOCAL 这个目录树可以给 TEX 系统 (并非指初始安装的系统，而是指随着时间推移你自己定制的一套) 放置一些和 TEX Live 版本无关的文件，主要是字体文件。默认值是 /usr/local/texlive/texmf-local，和具体 TEX Live 版本没有关联，因为同时这也是推荐你放置自己的本地宏包和配置的地方。

TEXMFSYSVAR 这个目录树用于给 texconfig-sys 命令放置版本**相关**的文件。默认值是 TEXDIR/texmf-var，通常不建议修改。因为还有一个变量是 TEXMFSYSCONFIG，给 texconfig 存放修改过的配置文件。参见第 4.1 节 (第 13 页)。

o -- 其他选项 在这个菜单里你可以选择下列选项：

a 指定另一个目录来存放生成的字体文件。前面提到，缺省是在 TEXMFVAR 目录树下。这个选项对于需要以只读方式挂载主目录树，同时又要在另一个位置 (通常依不同的主机而变化) 上存放动态生成的字体的情况比较有用。

l 在其他位置创建程序、man page 和 GNU Info 文件的符号链接。例如，你可能希望 man page 能放在 /usr/local/man 下，而 Info 文件放在 /usr/local/info 下边。(当然，你需要对这些目录有足够的权限才能在这些目录下创建链接。)

我们不建议你用这个选项来覆盖你的操作系统附带的 TEX 系统。它只是为了在用户已知的目录 (比如 /usr/local/bin) 中创建链接，而在此前这些目录里是没有任何 TEX 文件的。

d 跳过字体/宏包对应的文档的安装，这能节省不少硬盘空间，也有可能是因为你已经安装过了这类文档，为了避免冗余。

[1] 为了让你易于和安装界面对应，我们没有翻译这个列表。

s 跳过字体/宏包的源代码树的安装，这在你 (用 NFS 之类的方法) 通过网络和其他机器共享这个源代码树时会很有用。

i -- 执行安装 一旦你对当前的配置选项都满意了，就可以输入 i 来执行安装，这一步会把选定的内容从光盘复制到指定位置。

安装完成之后，下一步就是将 TEXDIR/bin 目录下的一个依机器架构而定的子目录添加到 PATH 中，这样新安装的这些程序可以直接使用。表 1 (第 8 页) 列出了所有的架构名称，你也可以看看 TEXDIR/bin 目录下有什么内容再决定。

上述步骤的具体命令、要使用的初始化文件，和你使用的 shell 环境有关。如果你使用的是 Bourne 兼容型 shell (sh, bash, ksh，等等)，可以将下面的内容放置到 $HOME/.profile 文件中：

```
PATH=/usr/local/texlive/2006/bin/archname:$PATH; export PATH
```

对于 C shell 兼容类 shell (csh、tcsh), 在 $HOME/.cshrc 文件中增加下面的内容：

```
setenv PATH /usr/local/texlive/2006/bin/archname:$PATH
```

主要的安装过程完成，又设置好环境变量之后，最后一步就是执行 texconfig 或者 texconfig-sys 来定制你的安装了。在第 4.1 节 (p. 13) 中对此有详细解释。

下面是一个注释了的小例子，其中只使用默认的目录，只安装针对当前系统的二进制文件。因此只需要用 i 安装即可。和前边一样，> 表示 shell 提示符。

```
> sh install-tl.sh
i                           # 执行安装
> texconfig ...             # 参阅第 4.1 节
# 设置新的 PATH 变量，以 Linux 为例:
> PATH=/usr/local/texlive/2006/bin/i386-linux:$PATH; export PATH
```

如果对上述步骤有何疑问，请咨询你那儿对系统配置熟悉的专家，看他能否帮你解决。比如如何挂载 TEX Live 光盘、安装到哪个或哪些目录、修改个人初始化文件的正确方法等等，都有可能依系统配置的不同而有区别。

3.2.1 非交互式安装

你可以使用环境变量来设置默认目录，从而实现在非交互环境下的安装。例如：

```
> TEXLIVE_INSTALL_PREFIX=/opt/texlive
> export TEXLIVE_INSTALL_PREFIX
> echo i | sh install-tl.sh
```

这里用 TEXLIVE_INSTALL_PREFIX 环境变量覆盖了默认的 /usr/local/texlive 目录，而其他选项不变。所以 TEX Live 在上述情况下会被安装到 /opt/texlive/2006。

和其他 Unix 下的程序一样，最后的 echo i 可以替换为任何其他能够直接输入的命令序列，因此这些命令也可以通过脚本方式来组合。

下面是所有可以设置的变量：

TEXLIVE_INSTALL_PREFIX 覆盖 /usr/local/texlive。

TEXLIVE_INSTALL_TEXDIR 覆盖 $TEXLIVE_INSTALL_PREFIX/2006。

`TEXLIVE_INSTALL_TEXMFLOCAL` 覆盖 `$TEXLIVE_INSTALL_PREFIX/texmf-var`。

`TEXLIVE_INSTALL_TEXMFSYSVAR` 覆盖 `$TEXLIVE_INSTALL_TEXDIR/texmf-var`。

`TEXLIVE_INSTALL_TEXMFHOME` 覆盖 `$HOME/texmf`。

也许我们更应该使用标准 GNU 风格的 `configure` 脚本来设置参数，而不是通过环境变量。欢迎志愿者帮我们改进!

3.3 将单独的软件包安装到硬盘

你可以将单独的软件包或集合从当前发行版安装到现有的非 TEX Live 安装中，或者安装到以前的 TEX Live 版本里。

以启用 Rock Ridge 扩展的方式挂载好 CD 或 DVD 之后，就可以开始使用了。具体的命令可能依系统的不同而有所不同，下面的例子在 Linux 下可以正常工作，不过设备的名称 (此处是 `/dev/cdrom`) 可能会有区别。(所有我们的例子都使用 `>` 作为 shell 提示符；所有用户的输入都有下划线。)

```
> mount -t iso9660 /dev/cdrom /mnt/cdrom
```

将当前目录更改为挂载点:

```
> cd /mnt/cdrom
```

Mac OS X 系统通常在 `/Volumes` 目录下列出自动挂载好的光盘。

执行 `install-pkg.sh` 安装脚本 (而不是用于完整安装的 `install-tl.sh`):

```
> sh install-pkg.sh options
```

下面这几个选项控制载入的内容:

`--package=`*pkgname* 需要安装的独立软件包。

`--collection=`*colname* 需要安装的独立的软件集合。

`--nodoc` 在执行操作时忽略文档文件。

`--nosrc` 在执行操作时忽略源代码文件。

`--cddir=`*dir* 读入的来源目录，默认是当前目录。如果你是按照上述步骤进行的，这个目录就是发行版的目录，不需要修改。

`--listdir=`*dir* 用于读入软件包信息的 `list' 目录，这个目录在 *cddir* 下。只在你希望修改 TEX Live 的时候才要设置这个选项。

而实际的操作是由下列选项设置的，如果这些选项都没有指定，缺省的操作是安装选定的文件。目的 (destination) 目录树是用 kpsewhich 展开 `$TEXMFMAIN` 变量得到的。你可以设置 `TEXMFMAIN` 或 `TEXMF` 环境变量来覆盖这个路径。

`--listonly` 仅列出需要安装的文件，而不执行实际的安装。

`--archive=`*tarfile* 不将文件安装到 TEX 系统中，而改为创建一个 tar 归档文件。

其他选项:

--config 安装后，执行 texconfig init。
--nohash 安装后，不执行用于重新构建文件名数据库的 mktexlsr 命令。
--verbose 打印更多脚本执行过程中的信息。

下面是一些使用的实例：

1. 列出而不安装 fancyhdr 宏包中的文件：

```
> sh install-pkg.sh --package=fancyhdr --listonly

texmf/doc/latex/fancyhdr/README
texmf/doc/latex/fancyhdr/fancyhdr.pdf
...
```

2. 安装 LaTeX 宏包 natbib：

```
> sh install-pkg.sh --package=natbib
```

3. 安装 LaTeX 宏包 alg，但不安装它的源文件和文档：

```
> sh install-pkg.sh  --package=alg --nosrc --nodoc
```

4. 安装所有额外 plain TeX 宏文件的集合：

```
> sh install-pkg.sh --collection=tex-plainextra
```

5. 将所有 pstricks 宏包的内容打包到 /tmp 目录下的 tar 文件里：

```
> sh install-pkg.sh --package=pstricks --archive=/tmp/pstricks.tar
```

如果对上述步骤有何疑问，请咨询你那儿对系统配置熟悉的专家，看他能否帮你解决。比如如何挂载 TeX Live 光盘、安装到哪个或哪些目录、修改个人初始化文件的正确方法等等，都有可能依系统配置的不同而有区别。

4 安装后的步骤

在主要的安装完成之后，剩余的步骤就是根据个人需要来配置 TeX 系统，使之符合你的需要，并执行一些基本的测试。

另一类安装后的工作是获得 TeX Live 中并不包含的的宏包、字体和程序。一般只需要将此类附加文件安装到 TEXMFLOCAL 目录树 (对于硬盘安装) 下，或者 TEXMFSYSVAR 目录树 (对于光盘执行) 下。参见第 10 页的``安装路径''选项。

不幸的是，针对不同软件的具体安装细节可能有巨大的差别，因而我们无法在此详述。这里有一些到相关叙述的外部链接：

- http://www.tex.ac.uk/cgi-bin/texfaq2html?label=instpackages
- http://www.ctan.org/installationadvice
- http://www.ctan.org/tex-archive/info/beginlatex/html/chapter5.html#pkginst
- http://www.ctan.org/tex-archive/info/Type1fonts 专门针对字体的安装信息。

4.1 texconfig 程序的使用

你可以，也应该在安装后使用 texconfig 命令来配置这套系统，使之符合你的需要。这个命令和其他程序一样，安装在依架构变化的 `TEXDIR/bin/`*arch* 子目录下。

如果你以不带参数的形式执行，它会进入全屏模式，允许你用交互的方式查看并修改选项。

你也可以带上各种命令行选项执行。下面是最常见的几个：

`texconfig paper letter` 将不同的程序或 driver (pdftex、 dvips、dvipdfm、xdvi) 默认的纸张大小设置为 US letter。其他可选的大小有缺省的 `a4` 等。

`texconfig rehash` 在添加或删除文件后更新所有 TEX 的``文件名数据库''。

`texconfig faq` 显示 teTEX FAQ。(参见 `texmf-doc/doc/english/FAQ-en` 上的主 TEX FAQ。)

`texconfig help` 输出 texconfig 的帮助信息。

当然 texconfig 只能修改 TEX 系统里大量选项和参数中的一部分。对基本 Web2C 程序通用的主配置文件是 `texmf.cnf`，你可以使用 `` `kpsewhich texmf.cnf' `` 找到它的位置，这个文件里的注释说明了缺省的设置和可选的参数。

texconfig 修改的是当前用户目录 (`$HOME/.texlive2006`) 下的配置。如果你安装的 TEX 只给自己一个人用，那这么做没什么不妥。不过如果 TEX 是安装在一个多用户的系统里，你可能希望更改整个系统的配置，这样的话运行的应该是 texconfig-sys 而非 texconfig。

与之类似，updmap 和 fmtutil 脚本也已改变为只修改 `$HOME/.texliveYYYY` 目录下的配置。请使用 updmap-sys 和 fmtutil-sys 来调整系统目录下的配置。

尤其需要注意的一点是，`fmtutil-sys --missing` 命令是很有用的 --- 预先在系统路径下生成好所有标准的格式，否则每个用户都得在自己的目录下生成相同的格式文件了，麻烦而且浪费空间。

此外，如果你自己修改过 `fmtutil.cnf` 或 `updmap.cfg` 而不用系统安装时附带的版本，那这些文件应该存放在 `TEXMFSYSCONFIG` 变量对应的那个目录树中。

第 2.3 节 (p. 5) 的环境变量里列出了所有这些命令可以更改的目录。你可以执行 `texconfig conf` 来查看当前配置的实际路径，也可以通过编辑 `texmf.cnf` 文件来修改它们。

4.2 测试安装是否成功

在完成你所需要的 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。

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 3.141592 (Web2C 7.5.5)
   kpathsea version 3.5.5
   ...
   ```

   如果返回的结果是 `command not found' 而非版本和版权信息，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 8 页关于设置环境变量的说明。

2. 处理一个基本的 LATEX 文件：

```
> latex sample2e.tex
This is pdfeTeXk, Version 3.141592...
...
Output written on sample2e.dvi (3 pages, 7496 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。你可以让 TEX 报告具体搜索的路径，以便仔细分析出错的原因。参见第 33 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
   ```

   (Windows 下类似的命令叫 dviout。) 你应该可以看到在新窗口中出现了一篇介绍 LATEX 基础的有趣文档。(如果你刚接触 TEX 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，如果不在运行或 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
   ```

   不幸的是 gv 和 xpdf 现在都不包含在 TEX 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。

7. 其他可能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 TEX 打印字体表格。
   story.tex 最经典的 (plain) TEX 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。

如果你还是个 TEX 新手，或者在编辑 TEX 或 LATEX 文档时需要帮助，请访问 http://tug.org/begin.html 寻找引导性的资源。

5 Mac OS X 下的安装

Mac OS X 下推荐的 TEX 安装方式是使用 2005 年新出现的 MacTEX 发行版。我们在 live DVD 的 `mactex/` 目录下提供了。它包括一套自带的 Mac OS X 风格的安装程序，可用于安装完整的 TEX 系统，这套发行版混合了 teTEX 和 TEX Live，并增加了不少额外的应用程序和文档。项目的主页在 `http://tug.org/mactex`。

如果你愿意的话，在 Mac OS X 下同样可以直接从 TEX Live 安装 TEX，下面介绍了使用 `install*` 安装脚本的方法。

你必须安装了 bash shell 才能在 Mac OS X 下执行安装脚本。如果你运行的是 Mac OS X 10.2 或者以后的版本，那已经符合要求了。但如果用的是更早版本的 Mac OS X，默认的 zsh shell 是不行的，必须从 Internet 上获取并安装 bash，或者更该考虑一下升级你的系统。

只要装好了 bash，就可以按照上述章节的 Unix 安装文档执行。参见第 6 页的第 3 节。上面必要的地方已经加上针对 Mac OS X 的说明了。

6 Windows 下的安装

在 TEX Live 的这个版本里，值得高兴的是，我们的发行版重新附带了一个专门针对 Windows 平台的安装程序，称为 tlpmgui.exe。(对于不同版本发行版的介绍，参见第 4 页的第 2.1 节。)

tlpmgui 提供和 Unix 安装程序相同的选项，不过是以图形用户界面的形式：一样可以选择安装方案、单独的软件集合、安装的目录，等等。第 3.2 节 (p. 8) 介绍了这些基本概念。tlpmgui 还支持添加/删除宏包、更新文件名数据库和构建格式文件这些安装后的工作。此外，tlpmgui 也能将系统配置为支持直接从 DVD 上运行。

喜欢究根问底的人也许有兴趣知道，tlpmgui 只是一个图形化的前端，其内部引擎是命令行程序 tlpm。

TEX Live 中 Windows 下的 TEX 系统中的二进制程序是 W32TEX 发行版提供的，感谢 Akira Kakuto 的支持。系统中还包括其他一些比较旧 (但依然可以正常工作) 的工具，这些工具是 Fabrice Popineau 制作的。以及一个新的 `dvi` 预览软件，Toshio Oshima 的 dviout。

TEX Live 可以安装在 Windows 98 / ME / NT / 2K 或 XP 下。不支持更老的系统，如 Windows (3.1x) 和 MS-DOS。

警告： Win9.x 的用户必须在安装前确保他们有足够的空间存放环境变量。 tlpmgui.exe 程序不会自动修改这个大小。由于在安装过程中需要设置一些环境变量，你可能会遇到环境变量空间不够的情况。可以把 `SHELL=<path>COMMAND.COM /E:4096 /P` 添加到你的 `config.sys` 文件中以增加这个空间的大小。

6.1 将 TEX Live 安装到硬盘

将 TEX Live CD 插入光盘驱动器后，应该就会自动运行 tlpmgui。如果没有，请点击 开始→运行，然后输入 `<drive letter>:\setuptl\tplmgui.exe` (如果是使用的是 TEX Collection DVD 则应该输入 `<drive letter>:\texlive\setuptl\tplmgui.exe`)。其中 `<drive letter>` 是 TEX Live CD (或 TEX Collection DVD) 对应的驱动器符号。然后点击确定。

一个标题为 TeX Live installation and maintenance utility 的安装窗口将会出现。它包含下面这几部分：Main customization (主要定制)、Install (安装)、Select a scheme (选择安装方案)、Select systems (选择系统)、Directories (目录) 和 Options (选项)。

安装程序所在的驱动器 (或目录) 将会出现在 Directories 里的 CD/DVD 按钮旁边 (例如 `F:/`

或 DVD 上的 F:/texlive/)。如果没有出现的话，请点击 CD/DVD 按钮，选择对应 TEX Live CD (或 TEX Collection DVD) 的那个驱动器。

点击 TLroot 按钮来设置希望安装到的目录。为了后面使用方便，TLroot 环境变量将被设置为这个目录。

在 Select a scheme 部分，你应该点击单选按钮来选择安装方案 (比如 scheme-gust)。每个方案旁都附带一个 Info 按钮，点击后可以显示它的简短说明。

``方案''是大量某种特定用途的文件的集合。基本、中等和完整安装都有其对应的方案。还有一些则是面向某个具体的 TEX 用户组织的 (比如 GUST 或 GUTenberg 推荐其会员使用的) 或专为某些应用设计的 (比如专门给 XML 和 TEX 协作而定制)。通过在 Main customization 部分的 Standard collections 或 Language collections 里选择其他的集合 (collection)，预设的方案可以进一步细化。比如点击 Standard collections 标签旁的 Select 按钮，MetaPost、Omega 和多种语言的文档这些集合都可以添加进去。

默认选择的 Wintools 集合包含许多 Windows 不会自带的重要小工具，它们会很有用，比如图形转换程序 sam2p, jpeg2ps, tiff2png；压缩/解压缩程序 bzip2, gzip, unzip 和用于新的 getnonfreefonts 工具的 wget 程序。

下一步，在 Main customization 部分点击标有 Language Collections 的 Select 按钮，这将打开 Language collections 窗口。在这个窗口中通过在语言旁边的复选框上打勾来选择安装语言。

接下来，点击位于 Install 部分的 Install 按钮开始相应的安装过程。

注意： Ghostscript、Perl 和 Wintools 都是缺省选定的，除非你已经安装过这些软件，或者明确知道自己在干什么，否则不应该取消选择，因为许多重要的工具都依赖它们。这还会使 PERL5LIB 和 GS_LIB 环境变量得到设置或修改。

TEX Live 系统在安装后还需要一些步骤 (格式文件的生成、ls-R 数据库的生成、设置环境变量，等等)。所有这些操作均在此时完成，而且其中有些过程可能比较耗时，所以请耐心等待，直到你看到关于成功完成安装的提示出现。

tlpmgui 的快捷方式将被添加到 开始→程序→TeXLive2006 菜单中 (dviout 如果安装了，也会添加快捷方式)。

在必要时 (Win9x/WinME)，安装程序会要求你重启计算机。

6.2 为 Windows 提供的支持性软件包

为了完成 TEX Live 的安装，需要补充一些 Windows 系统下通常没有的软件包。许多脚本是使用 Perl 语言写成的。一些重要的工具程序需要 Ghostscript PostScript 解释器来渲染或转换文件。有时还需要一个图形文件工具包。最后，同样重要的是，一个针对 TEX 的编辑器能让你更轻松地输入 TEX 文件。

尽管不难找到这些工具的 Windows 版本，但为了节省你的力气，我们在 TEX Live 中包括了这样一些工具：

- GNU Ghostscript 8.54
- 最小化的 Perl 5.8，但足以运行 TEX Live 下所有的 Perl 脚本
- 一系列小程序的集合 win-tools (bzip2、gzip、jpeg2ps 和 tiff2png)

Perl 和 Ghostscript 是缺省已选定的安装套件，如果你已经装过了，也可以取消选择。

如果你不愿意安装这个套件，也可以自行安装需要的工具以完善你的 TEX Live 系统。下面列出了这些软件的来源：

Ghostscript http://www.cs.wisc.edu/~ghost/

Perl http://www.activestate.com/ 不过你可能需要从 CPAN (http://www.cpan.org/) 获得一些辅助性软件包。

ImageMagick http://www.imagemagick.com

NetPBM 你还可以用 NetPBM 来替代 ImageMagick 的图形文件处理和转换功能。NetPBM 的主页在 http://netpbm.sourceforge.net/

针对 TEX 的编辑器 有很多选择，通常只是个人偏好问题。下面是一些选项：

- GNU Emacs 在 Windows 下有 native 的版本，参见 http://www.gnu.org/software/emacs/windows/ntemacs.html
- Emacs 的 AucTeX 套件在 Windows 下有 native 的版本，参见 ftp://alpha.gnu.org/gnu/auctex
- WinShell 可以在 TEX Live 的 support 目录下找到，参见 http://www.winshell.de
- WinEdt 是 http://www.winedt.com 提供的一个共享软件
- Vim 可以在 http://www.vim.org 得到。
- TeXnicCenter 是一款自由软件，由 http://www.toolscenter.org 提供，在 proTeXt 发行版中也能找到。
- LEd 可以在 http://www.ctan.org/support/LEd 找到
- SciTE 可以在 http://www.scintilla.org/SciTE.html 找到

你也许想安装其它非自由[2]的工具，但 TEX Live 中并没有包括它们，如 GSView，Ghostscript 的伴侣，可以用它更方便地阅览 PS/PDF 文件。GSView 可以从 http://www.cs.wisc.edu/~ghost/gsview/ 或者任何一个 CTAN 站点获得。

7 维护 Windows 下的安装

如果你已经安装好了 TEX Live，就可以再次运行 tlpmgui 来修改和维护你的安装了。

7.1 添加/删除软件包

因为 tlpmgui 的快捷方式位于 开始→程序→TeXLive2006 菜单内，从这个地方启动它。一个标题为 TeX Live installation and maintenance utility 的窗口将出现。它包括四个选项卡：Add Packages、Remove packages、Manage installation、Remove installation。

点击标有 Add packages 或 Remove packages 的选项卡进入相应的功能：

- 在第一个选项卡中单击 CD/DVD 按钮选择正确的 CD 驱动器 (或带有 texlive 目录的 DVD)。
- 在 Buttons 部分，点击 Search 按钮显示要安装或删除的软件包列表，或在 Select packages to... 部分刷新它。

  当安装软件包时，将比较已经安装的和 CD/DVD 上存在的软件包，列出未安装的那些。然后，根据你的喜好来选择需要安装的包。

  当删除单个软件包时，只显示已安装的软件包的列表。

  需要注意，对于 Add packages 和 Remove packages，首先列出的是软件集合。

[2]也就是说你不能根据自由软件准则自由修改或重新发布，但这并不意味着你不能免费获得它们。

- 通过单击软件包的名称选择它。点击 Buttons 部分的 Info 按钮后，位于 Info about the selected item 部分的窗口中将显示选定包的简单描述。要选择多个软件包，可以在按下鼠标左键的同时按下一个 Ctrl 或 Shift 键，或者在拖动鼠标指针时按下左键。
- 在 Buttons 部分点击 Install 或 Remove 按钮以执行相应的操作。

7.2 配置和其他管理任务

这个功能在 Manage the installation 标签下提供，如果你希望在安装后添加对某个语言的支持，或者添加某个一开始安装时没有选择的格式文件，又或者是要重新生成某个修改过的格式，它会很有用。

你可以执行下列操作：

- 刷新 `ls-R` 数据库
- 创建格式文件 (所有的或者缺失的)
- 编辑 `language.dat`
- 编辑 `fmtutil.cnf`
- 编辑 `updmap.cfg`

注意：关闭 Edit... 窗口后会提示你选择 Cancel 还是 Done，如果选择后者，将重新生成格式文件 (如果你编辑的是 `updmap.cfg`，则重新生成字体的 map 文件)，然后刷新 `ls-R` 数据库。

关于配置过程的更多信息请参见第 7.8 节 (p. 20)。

7.3 卸载 TeX Live

标着 Remove the TeX Live installation 的选项卡打开的窗口，功能不值得描述。我们既不知道谁会需要，也不知道它为何存在 ...:-)

然而，如果你在 `texmf-local` 目录存放了自己添加的东西，删除程序将不会清除它，也不会删除该目录下的东西。存放 tlpmgui 和相关文件的 `setup-win32` 目录也不会被删除。你将不得不手动清除它们。

7.4 将你自己的软件包添加到安装路径中

首先，无论你更改了什么，**别忘了重建 ls-R 数据库**。否则，你的新文件将不会被发现。你可以运行 tlpmgui，从 Manage the installation 标签下选择适当的操作，或者手动运行 `mktexlsr` 命令来重建该数据库文件。

如果你想添加 TeX Live 发行版没有提供的文件，推荐的存放位置是 `$TEXMFLOCAL` 目录。在升级 TeX Live 软件时，这种方式是安全的。

最初，`$TEXMFLOCAL` 指向的目录是空的。比如你希望增加 Maple 符号计算程序的支持文件，你应该将样式文件放在：

```
c:\TeXLive2006\texmf-local\tex\latex\maple\
```

而相应的文档文件放在：

```
c:\TeXLive2006\texmf-local\doc\latex\maple\
```

7.5 在命令行环境下执行 `tlmp.exe`

作为 tlpmgui 引擎的 tlpm.exe 程序有许多有用的选项，可以运行下述命令得到其列表：

```
tlpm --help
```

你可以在 `tlpm.readme` 中找到更多的说明和举例。

7.6 网络安装

Kpathsea 支持 UNC 名称，所以你能够用它从网络上获取 TEXMF 目录树。但还有更好的办法：除了 `bin/win32` 目录下的文件以外，所有的支持文件和配置文件都可以与 teTEX 或 Unix 下的 TEX Live 共享。这意味着，你能够通过使用 Samba 从一台 NT 服务器挂载到一台 Unix 工作站上或反过来。有数种可能的策略可供选择：

- 将所有的东西放在服务器上。仅仅加入你想在 `bin` 目录下使用的 OS 或架构特有的文件，例如 `bin/win32` 和 `bin/i386-linux`。随后配置你的主要变量。你可以使用 UNC 名称指向 Win32 下正确的目录。
- 安装二进制和格式文件的本地复本。在这种情况下，将 `$TEXMFMAIN` 指向将被远程接入的主 `texmf` 目录树。将 `$TEXMFVAR` 指向一个本地目录，放置本地配置文件和实时生成的文件。

7.7 Windows 版本与其它版本的区别

需要指出的是，Windows 版本的 Web2C 有一些和其他系统下不同的特性。

kpsecheck 这一命令提供了一些不能很好地被包括到 kpsewhich 中的选项。它允许你列出 texmf 目录树中多次出现的文件。这是非常方便的，但有时你也会得到不想得到的结果 (如大量的 `README` 文件)。需要注意的是，这些文件可能会导致 Kpathsea 散列机制的内部冲突；幸运的是，Kpathsea 从不寻找这些文件。正因为这一原因，你可以将 `-multiple-occurrences` 与两个其他包括或排除任何文件名的选项结合起来。文件名可以用一些模式进行匹配 (你可以指定数个模式)。

kpsecheck 命令也将报告共享内存的状态：使用中或没有使用。了解这一情况可能是有用的，因为如果状态显示是 `使用中'，意味着一个或多个进程正在工作，此时执行任何 mktexlsr 命令都不会生效，直到正在运行的这些链接了 Kpathsea 的进程都退出为止。

最后，这个命令还能报告它找到的 Ghostscript 的位置。在 Win32 下，对于许多其他的程序来说，通过 Ghostscript 的注册表项找到并使用 dll 比改 `PATH` 更为简便，毕竟 `PATH` 长度是有限的。

Web2C 与 Unix 下的 Web2C 相比，该引擎支持的选项要多一个：

- `-proctimes` 打印出有关作业运行时间的统计。需要说明的是 Win9x 不能被称为一个真正多任务操作系统，对于较短的时间段，它不能提供可靠的记时器，因此，打印出来的数值只是近似值。在 NT/2K/XP 下，作业运行过程中的用户时间和系统时间的结果相当精确。Unix 用户请注意：Windows 下的 `time` 命令和 Unix 下的含义完全不同。

7.8 个人定制

7.8.1 Dvips

dvips 的配置文件在:
`C:\TeXLive2006\texmf-var\dvips\config\config.ps`
你可以使用任何编译器将其打开并且更改某些参数:

fonts 如果 dvips 需要生成 PK 字体，你可以更改默认打印机的 METAFONT 模式或者分辨率。默认情况下，dvips 使用 Type 1 版本的 CM 字体，因此，它很少用到 mktexpk;

printer 可以告诉 dvips 默认情况下你想在什么地方打印。如果 `o` 选项后没有紧接着打印机名的话，将会生成 `.ps`，即 PostScript 文件。你可以以如下方式为 dvips 设置打印机:

```
o lpt1:
% o | lpr -S server -P myprinter
% o \\server\myprinter
```

paper 接下来，你可能希望改变纸张的大小，比如从欧洲规格 (A4) 改至美国规格 (letter)，这得把该文件中首个指定的纸张大小设为 letter。在配置文件里找到下面这段，把它移动到所有以 `@` 开头的行之前:

```
@ letterSize 8.5in 11in
@ letter 8.5in 11in
@+ %%BeginPaperSize: Letter
@+ letter
@+ %%EndPaperSize
```

当前的 TEX Live 发行版已经通过程序为 Dvips 和 Pdftex 提供了正确更新的字体映射文件。这是用 updmap 程序在安装后完成的。如果你用安装程序或者手动添加了新的字体包，就应该自行修改 `$TEXMFVAR/web2c` 目录下的 `updmap.cfg` 文件。

7.8.2 PdfTeX

如果你用 pdflatex 程序来直接输出 PDF 文件，同时也想使用 US letter 尺寸的纸张，就该编辑 `C:\TeXLive2006\texmf-var\tex\generic\config\pdftexconfig.tex`，把 `` `\pdfpagewidth' `` 和 `` `\pdfpageheight' `` 的定义作如下改变:

```
\pdfpagewidth=8.5 true in
\pdfpageheight=11 true in
```

保存文件，然后退出编辑器。

7.8.3 XeTeX

XeTEX 的 Windows 版本是与 fontconfig 2.4.2 静态链接的。fontconfig 的配置文件之一是 `fonts.conf`。请参见 `fonts.conf` 文件本身以了解其细节，这个文件在命令 kpsewhich --var-value=FONTCONFIG_PATH 输出的目录下。

7.8.4 GSView

GSView 现在在 Aladdin 许可证下发布，因此 TEX Live 不再包括此程序。

如果你想把纸张大小改为 letter，从 开始→程序打开 GSView，选择 Media→Letter即可。

另外还可以修改菜单设置来改进屏幕图像的可读 (readable) 性。在 Media→Display Settings 下，将 `Text Alpha` 和 `Graphics Alpha` 设为 4 bits。

注意，安装程序已经将所有的 `.ps` 和 `.eps` 文件自动关联到 GSView 程序。

至于打印的步骤，参见下面的第 7.10 节。

7.9 测试

对于一般性的测试程序，请参见第 4.2 节 (p. 14)。这里介绍的只是针对 Windows 的测试。

在你的编辑器 (Xemacs, WinShell) 中打开 `C:\TeXLive2006\texmf-dist\tex\latex\base` 下的 `sample2e.tex` 文件。屏幕上应该会出现一个 LATEX 源文件。点击 Command→LaTeX 菜单 (XEmacs) 或工具栏上的 LATEX 图标 (WinShell) 可以对其进行编译, 然后点击 Command→View DVI 菜单 (XEmacs) 或 Preview (dviout) 图标 (WinShell) 预览生成的 .dvi。

在第一次用 dviout 预览的时候将生成一些字体，因为用于屏幕预览的字体没有安装。一会儿之后就已经生成好了大多数你用到的字体。此后你就很少再看到生成字体的窗口了。

对日后的提示：如果运行的 LATEX 因为找不到某一文件而停止了，你可以按 Ctrl-z 退出。

7.10 打印

可以用 dviout 来直接打印，此时打印工作交给 Windows 统一打印驱动完成，理论上它和所有的打印机都兼容。然而，这也存在一些缺点，比如它会生成巨大的假脱机 (spool) 文件，或者某些较早版本的 Windows 对统一驱动的支持不太好。它的优点是你能用到嵌入 BMP 或 WMF 图像的特性。你也必须正确设置打印机参数，否则，打印出来的效果是缩放后的 (在 300 dpi 打印机上以 600 dpi 打印会使得一页纸上仅显示四分之一的页面)。

如果你使用 dvips 生成 `.ps` 文件，然后用 GSView 打印的话，打印过程将会快很多且更加可靠。在 GSview 中，选择 File→Print...，将出现 Print 窗口。

如果用的是 PostScript 打印机，**一定要选择** *PostScript Printer*。这一过程可以在 Print 窗口左侧底部的 Print Method 框中完成。然后就能选择任何一台你已事先安装好的打印机。如果没有选中 `PostScript Printer` 复选框，是没法打印的。

如果你使用你自己的非-PostScript 打印机，则要在 Print Method 框中选择 Ghostscript device，然后，点击标有 djet500 的按钮，从弹出的窗口中选择你的打印机型号。(在更旧版本的 GSView 中，确定 `PostScript Printer` **没有**被选择，然后从 Device 列表中选择打印机型号。)

7.11 Win32 下的使用技巧与窍门

7.11.1 Win32 的不同种类

我们所谓的 Win32 其实并非指这个操作系统本身，而是指一系列函数的集合 (微软的 SDK 头文件里包括了约 12,000 个这样的函数)，你可以使用这些函数为 Windows 家族中的不同操作系统编写程序。

Windows 有多个版本:

- Win95、Win98 以及 WinME，它们**不是真正的多任务、多线程**环境，只是 DOS 的更新变体。这可以由它们在启动时，PC 装载 `command.com` 解释器得到证明。如果你在这里停止启动过程，你可以查询当前 (DOS) 的版本，它将会回答你类似 `` `MS-DOS 7.0' `` 之类的答案 (至少在旧版本的 Win9x 上是这样的)。
- Windows NT，是一套重新编写的操作系统，支持真正的多任务，并包括许多高级的特征。
- Windows 2000 基于 NT，但具有 Win98 所有的引入注意的修饰 (bells and whistles)。
- Windows XP 有两个不同的版本：个人版与专业版。这是融合两条不同路径上的产品 (基于 Win9x 和基于 NT) 的最后一步。XP 基于 NT。

Win9x 支持并行支持 32 位程序和 16 位程序。但操作系统本身并非完全工作在 32 位模式下，也不支持内存保护：16 位程序甚至可以覆盖操作系统本身占用的内存！系统的某些部分，如 GDI (图形设备接口) 只能管理当前运行的所有程序中的有限的资源，如位图、字体、画笔等。由于位图头结构 (bitmap header) 这样的资源不允许超过 64KB，这样你能明白为什么大量使用图形对象时系统会非常吃力了。

NT、2K 和 XP 不受这些限制，也不受 Win9x 中其他条件的限制。它们是真正的多任务环境，并且具有内存保护机制。与 Win9x 相比，它们响应更快，因为具有更好的内存管理、更好的文件系统等等。

7.11.2 命令行提示符

你可能觉得奇怪，``我已经有 Windows 了，为什么还需要命令行提示符呢？''

这是个很好的问题，而原因也具有通用的性质：并非所有操作都只用 GUI 就能解决。命令行给予了你编程的能力--- 在命令行解释器足够聪明的情况下。

但这个问题可能更加基础：TEX 是一个 **批处理** 工具，而不是一个交互工具。TEX 需要计算每一页的最佳布局，解决交叉引用等。这只能通过全局处理整个文档而达到。它不是一个可以通过交互操作而完成的任务。

这就是说，你应该以命令行方式来使用 TEX。实际上，这不算是很坏的选择。优势之一是可以为复杂的处理写出命令行工具：更容易调试，因为它们与任何 GUI 问题都不相关；而且 GUI 工具也可以用来作为命令行工具的界面。这就是为什么你大多时间内会通过 GUI 文本编辑器与 TEX 打交道的道理。

然而，在很多时假，你可能需要使用命令行提示符。其中的一种情形是你遇到困难，并且希望调试你的配置。

Win9x 你可以通过从开始→程序 菜单中选择 MS-DOS 图标或在 开始→运行 菜单中输入 `command.com` 来打开命令行提示符。

NT, 2K, XP 你可以在 开始→附件 菜单中选择 命令提示符 来打开命令行提示符。也可以选择 开始→运行，然后输入 `cmd.exe`。这是 NT 下全新的命令解释器 (因此，别把它叫做 *DOS* 窗口！)。

这里所提到的位置可能因 Windows 版本的不同而发生变化。

7.11.3 路径分隔符

尽管 Win32 API 可以正确解释作为路径分隔符的 `/` 和 `\`，但命令解释器却不行！因此，只要在编程用到路径名称时，你可以任意使用这两个分隔符之一，甚至在同一个路径中混用。但在命令行中就必须用 `\` 作为路径分隔符。原因是显而易见的，命令处理器使用 `` `/' `` 引入命令的参数。

最后，当看到采用 Unix 习惯写成的路径名不必惊讶。Windows-TEX Live 是 Web2C 的一个移植，其目标是跨平台兼容。正因为这个原因，所有的需要指定路径名称的配置文件都采用了 Unix 习惯。

7.11.4 文件系统

与 TEX 有关最差的 Win9x 的特征是所谓的 FAT 文件系统。TEX 使用了非常多的小文件，大小在 1--3kB 之间。FAT 文件系统相对较老，早于目前我们使用的大容量硬盘十数年。因此，它不能有效地管理 TEX Live 下数以万计的 TEX 文件。FAT 文件系统在一个巨大的分区中为**任何**一个文件分配至少 32kB 的空间。这表示，TEX 将占用比它实际需要多得多的磁盘空间。

其他的、更加现代的文件系统，如 FAT32 和 NTFS 不存在这些缺陷，它们仅管理 4kB 的簇。(你在 NTFS 系统上还能将这个限制降到 512 字节。)

7.11.5 如何将某些路径添加到你的 PATH 中

在操作系统中存在一些变量和值，它们的行为与你的程序中的全局变量比较类似。这些变量称为环境。任何程序在运行时需要环境的复本进行初始化。它可以请求或改变变量的值。这些改变只发生于复本的环境中，因此不会传递给其它正在运行的程序。

PATH 是一类特殊的环境变量，用于对你希望运行程序的搜索。Win9x、WinME 和 NT/2K/XP 下修改 PATH 的方法不尽相同。

Windows 95/98 编辑 `autoexec.bat`。在这个文件中，必然有一行以 `PATH=` 开头，随后是一系列以 ; 分隔的目录。请把含有可执行文件的目录加到这一行。然后，这一行看上去象这样:

```
PATH=c:\windows;c:\windows\system;c:\TeXLive2006\bin\win32
```

Windows ME 你需要运行一个特殊的命令 `c:\windows\system\msconfig.exe` 来改变任何环境变量。在这个程序中，选择 `环境' 选项卡，然后增加或修改你希望的变量。你需要重启机器使任何更改的生效。

Windows NT/2K/XP 点击 开始→设置→控制面板。现在，带有控制面板图标的窗口被打开。双击系统图标。系统属性窗口被打开。点击 高级 选项卡或者从对话框中寻找标有 环境变量 的按钮。现在你可以更改你的用户帐户的环境变量了。注意：那里同时还会显示系统的环境设置。通常，你不能更改系统变量，除非你有机器的管理户权限。如果你希望为所有用户更改 `PATH`，你必须通知系统管理员，或者你自己必须是系统管理员 --- 在后一种情况下，你得明确自己的操作没有错误。

如果那里已经有了你的帐户的 `PATH` 设置，双击 `PATH`，在变量名区域将出现 `PATH`，而区域的值显示的是以 ; 分隔的一系列目录表现出的当前设定。将含有可执行文件的目录加进去 (如 `c:\TeXLive2006\bin\win32`)。如果你的用户帐户下不存在这个变量，双击变量区域，输入 `PATH`，在变量值区域输入可执行文件所在的目录。注意：点击确定之前先点击应用按钮，否则对`PATH`的更改不会应用到你的系统中。在更改环境设置时一定要谨慎。

确定一个变量 VARIABLE 是否被正确设置的最好方法是打开终端，然后输入:

```
set VARIABLE
```

它应该会返回相应的值。

7.11.6 TEX 引擎

如果你已经看过 Web2C 的文档，你会发现所有基于 TEX 的不同程序都使用同样的基础引擎。例如，`tex.exe` 和 `latex.exe` 是相同程序的复本，但每个程序根据调用名称不同使用了不同的格式文件。

在 Unix 下，这种特性是通过 **符号链接** 实现的。 它可以节省一些磁盘空间，因为一些引擎使了很多不同的格式文件。

但是，Win32 API 不了解文件链接，因此，为了节省几乎相同的内存，所有的 TEX 基础引擎均被放在 DLL (**动态链接库**) 中。这表示，你将看到类似下面的输出：

```
2006-11-23  07:06             2 560 latex.exe
2007-01-06  23:54         1 073 664 pdftex.dll
2006-01-28  08:05             2 560 pdftex.exe
```

这里，`latex.exe` 仅仅是 `pdftex.exe` 的一个复本，使用了相同的核心文件 `pdftex.dll`。相同的技巧应用于 `mktex*.exe` 家族的程序，它们都链接到 `mktex.dll`。

实际上，一个通用的工具，`irun.exe` 可以用来为 Win32 下可执行文件生成与 Unix 下硬链接对等的文件。

7.12 如果遇到问题

7.12.1 如果 `latex` 没有找到你的文件，该怎么办？

- kpsewhich 可以用来调试任何问题。然而 kpsewhich 向标准错误 (stderr) 输出调试信息，而且旧版的 Windows 终端无法将重定向标准错误信息至某一文件。 (NT 和 Windows 2000 的终端知道如何做。但这一技巧可以应用到任何终端。) 为了诊断，你可以 (在 DOS/cmd 对话框下) 临时设定一个环境变量：

  ```
  SET KPATHSEA_DEBUG_OUTPUT=err.log
  ```

  也可以设定调试的级别：

  ```
  SET KPATHSEA_DEBUG=-1
  ```

  如果你希望把 stderr 重定向到 stdout：

  ```
  SET KPATHSEA_DEBUG_OUTPUT=con:
  ```

  通过这种方式，你可以在同一个文件中同时截获标准错误与标准输出。

- 假定系统被安装在 `c:/TeX`，检查下面的一些值：

  ```
  kpsewhich -expand-path $SELFAUTOPARENT   c:/TeX
  kpsewhich -expand-path $TEXMF            c:/TeX/texmf....
  kpsewhich -expand-path $TEXMFCNF         .;c:/TeX/texmf-var/web2c;
  kpsewhich -expand-var $TEXINPUTS         .;c:/TeX/texmf/tex//
  ```

- 如果其它 TEX 相关的值已经存在于你的环境变量设置中，请删除它们。它们覆盖了 texmf.cnf 中相应的变量。

- 检查从下面输出的值：

  ```
  kpsewhich cmr10.tfm   c:/TeX/texmf-dist/fonts/tfm/public/cm/cmr10.tfm
  kpsewhich latex.fmt   c:/TeX/texmf-var/web2c/latex.fmt
  ```

- 到此为止，如果所有的输出都是正确的，TEX 及其相关程序都应该可以正常工作了。否则你只好以 `-debug=n` 选项调用 `kpsewhich`，检查所有前面提到的值。试着分辩并报告问题。

7.12.2 如果你的安装还是不能如预期般工作，该怎么办？

这需要调查以下几个问题：

1. `tex.exe` 是否在我的 `PATH` 中？
2. `TEXMFCNF` 是否正确指向了 `c:/TeXLive2006/texmf-var/web2c` (默认值)？
3. 由 `tlpmgui.exe` 生成的记录文件中是否提示错误信息？ `tlpmgui.log` 可以在你的 `TEMP` 目录下找到。你可以通过搜索 `` `Error' `` 字符串定位错误信息。提示：记录文件可能会在生成所有格式文件后显示一些错误。不必惊慌：也许一些格式文件没有被安装。
4. 在 `http://tug.org/texlive/` 是否有相关的错误已被修正？(可能性不大，但不妨去查查)

TEX Live 软件包括了数百个程序以及数以万计的文件，所有的这些都有不同的来源。因此，对出现的问题预测所有可能的原因是相当困难的。不过我们还是会尽力帮助你的。(参见第 1.2 节，p. 3。)

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。

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。

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` 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。核心的 TEX 程序包括：

bibtex 维护参考文献。

dmp 从 troff 格式转换为 MPX (MetaPost 图片)。

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 proofsheet。

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。

makempx 排版 MetaPost label。

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。

mpto 展开 MetaPost 的 label。

newer 比较文件的修改时间。

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 TEX。

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 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 TEX 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 TEX 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 首先见扯是否有符合 p 的文件名数据库 (见第 30 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、再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 TEX 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。

尽管最简单也最常兼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。

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 33 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`[.`*progname*`] [=]` *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 `.*progname*' (程序名) 存在，则该定义只对正在运行的名叫 *progname* 或 *progname*`.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 28 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 -1LAR ./ >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重要的选项。

`--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*。默认情况下视同过文件名来猜测格式的。对于扩展名有二义性的格式，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
```

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 METAFONT 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` prolog 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。

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：

```
\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 1。

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 `.dvipsrc` (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.radicaleye.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：

```
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 1: 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 2: 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 3: 寻找字体文件

```
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。每修改一次都比续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 PICTEX 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 TEX 太难实现，才通过这些选项提供了一些灵活性。

9 致谢

TEX Live 是在几乎所有 TEX 用户组织的协力下完成的。这个版本由 Sebastian Rahtz 与 Karl Berry 负责监制。下面列出了主要的贡献者:

- 英国、德国、荷兰和波兰的 TEX 用户组织 (分别为 TUG, DANTE e.V., NTG, 和 GUST)，他们一同为所在地区的 TEX 社群提供了必备的技术和管理基础。请加入你本地的用户组织!
- CTAN 团队，他们负责分发 TEX Live 光盘镜像，为软件包更新提供支撑，TEX Live 正是基于这些软件包构建的。
- Peter Breitenlohner 和 ε-TEX 团队，他们创造了未来 TEX 的稳定基础。
- Thomas Esser，如果没有他优秀的 teTEX 套件，TEX Live 就不可能存在，在他坚持不懈地帮助下，TEX Live 得以不断改进。
- 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 TEX4ht 程序用于创建这份文档的 HTML 版本。
- Hans Hagen，他对 ConTEXt 格式做了许多测试和修改，使之更为适合 TEX Live 的要求。
- Hàn Thế Thành, Martin Schröder，和 pdfTEX 团队，他们持续不断地改进 TEX 的功能。
- Taco Hoekwater，他的开发使 MetaPost 和 TEX 本身焕发新的活力。

- Paweł Jackowski，他创建了 Windows 下的安装程序 tlpm，和 Tomasz Łuczak，因为他的 tlpmgui。
- Akira Kakuto，因为他在构建 Windows 下的二进制版本方面给予了我们巨大的帮助，现在的版本是在他的 W32TEX 发行版 (`http://www.fsci.fuk.kindai.ac.jp/kakuto/win32-ptex/`) 基础上完成的。
- Jonathan Kew 和他的雇主 SIL，因为 XeTEX 是如此新颖而重大的改进。他还花了大量时间和精力来将其集成到 TEX Live 中。
- Reinhard Kotucha，参与了大量更新 TEX Live 中软件的工作，以及 Windows 下的研究性工作，`getnonfreefonts` 脚本，等等。
- Petr Olšak，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。
- Fabrice Popineau，他创建了 TEX Live 最早的 Windows 支持。
- Norbert Preining，帮助构建 TEX Live 的基础结构，负责软件包的更新，并协调 Debian 版本的 TEX Live (与 Frank Küster 一起)，还一直提供了许多改进的建议。
- Staszek Wawrykiewicz，TEX 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，他耐心地组织和维护 Web2C，这是整套系统的基础。
- Gerben Wierda，他创建和维护 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目录。

二进制版本的编译者: Tigran Aivazian 和 Hartmut Henkel (x86_64-linux), Manfred Lotz (i386-freebsd), Fabrice Popineau (win32), Norbert Preining (alpha-linux), Vladimir Volovich (powerpc-aix, sparc-linux, sparc-solaris), Karl Berry (i386-linux), Olaf Weber (mips-irix), Gerben Wierda (i386-darwin, powerpc-darwin).

文档和翻译的更新: Karl Berry (英语), Daniel Flipo & Fabrice Popineau (法语), Günter Partosch & Hartmut Henkel (德语), Petr Sojka & Jan Busa (捷克语/斯洛伐克语), Boris Veytsman (俄语), Staszek Wawrykiewicz (波兰语).

当然，最重要的感谢应该致予 Donald Knuth，感谢他发明了 TEX，也感谢他将 TEX 赠与全世界。

10 发行历史

10.1 过去

1993 年末荷兰 TEX 用户组开始为 MS-DOS 用户开发 4AllTEX 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 4AllTEX CD，同时 TUG 技术委员会工作组也开始设计 ***TEX* 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 TEX 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。4AllTEX 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 TEX 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 teTEX 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 teTEX 的特性囊括在内，这样我们决定在 teTEX 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， teTEX 的一个新版本出现了，TEX 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 teTEX 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 TEX Live 5 的首要变化还是移除了所有的非自由软件。TEX 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 teTEX 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 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. 4)。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. 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 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，因为版本号增长得实在太快，现在简单地使用年份来标识版本：TEX 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 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. 4)。

- 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. 5) 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-doc/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-doc/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/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。)
- 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 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/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/` 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` 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 `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-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其他的地方了。
- 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)。参见第 13 页的第 4.1 节。
- 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。

10.2 现状

2006--2007 年，TEX Live 的一个重大变化是增加了 XeTEX，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`)，pdfTEX 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。)

安装脚本现在可以接受一些环境变量，以实现非交互环境下的安装。参见第 3.2.1 节。

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 teTEX (`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 teTEX 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。

10.3 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。如果你愿意帮助，请毫不犹豫地告诉我们!

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 TEX 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。

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