```
% $Id$
% TeX Live documentation.  Originally written by Sebastian Rahtz and
% Michel Goossens, now maintained by Karl Berry and others.
% Translated by Jjgod Jiang  <gzjjgod@gmail.com>,
%               Jinsong Zhao <jinsong.zhao@gmail.com>,
%               Yue Wang     <yulewang@gmail.com>
%               Helin Gai    <helin.gai@gmail.com>
% Public domain.
%
\documentclass{article}
\let\tldocenglish=1  % for live4ht.cfg

\usepackage{tex-live-zh-cn, indentfirst}

\title{%
  {\huge \textit{\TeX\ Live 指南---2016}}%
}

\author{Karl Berry 编写 \\[3mm]
        \url{http://tug.org/texlive/}
       }

\date{2016 年 6 月}

\begin{document}
\maketitle

\begin{center}
今年的 \TeX\ Live 谨献给我们不幸离世的同事 Sebastian Rahtz 和 Peter Breitenlohner。
\end{center}

\begin{multicols}{2}
\tableofcontents
%\listoftables
\end{multicols}

\section{简介}
\label{sec:intro}

\subsection{\TeX\ Live 与 \TeX\ Collection}

本文档描述 \TL{} 软件的主要功能和特性，\TL{} 是 \TeX{} 及其相关程序在
GNU/Linux 及其他类 Unix 系统、\MacOSX\ 和 Windows
系统下的一套发行版。

你可以直接下载 \TL{}，也可以在 \TeX{} 用户组织给会员分发的 \TK{}
\DVD 中找到。第~\ref{sec:tl-coll-dists}~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 \TL{} 本身。

\TL{} 包括了 \TeX{}, \LaTeXe{}, \ConTeXt, \MF, \MP, \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。

文档末尾的第~\ref{sec:history}~节 (第~\pageref{sec:history}~页)
介绍了这一版 \TL{} 的重要改变。

\htmlanchor{platforms}
```

\subsection{操作系统支持}
\label{sec:os-support}

\TL{} 为多种基于 Unix 的平台提供了可执行文件，包括 \GNU/Linux、
\MacOSX{}、和 Cygwin。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，\TL{} 仅支持 Windows Vista 或后续版本。
Windows~XP 和 2000 可能可以继续工作。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 \TL{} 以外，Windows 和 \MacOSX 用户还有其它的选择，请参考
第~\ref{sec:tl-coll-dists}~节。

\subsection{\protect\TL{} 的基本安装}
\label{sec:basic}

你可以使用 \DVD{} 方式或者网络方式来安装 \TL{}
(\url{http://tug.org/texlive/acquire.html})。
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 \TL{} 一小部分的用户来说非常适宜。

\DVD{} 安装程序可以把 \TL{} 安装到你的本地磁盘上。你不能直接从 \TK{}
\DVD{} (或者它的 \code{.iso} 镜像) 上运行 \TL{}，但你可以在一个 \USB{} 盘上
安装一套可以运行的版本 (参见 \ref{sec:portable-tl})。安装方法将
在下面的章节介绍 (\p.\pageref{sec:install})，这里提供一个快速入门:

\begin{itemize*}

\item 安装脚本的名称是 \filename{install-tl}。它可以在指定了
\code{-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 \code{-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 \code{-gui=perltk} 选项启用。

\item 安装完成后可以得到一个名为 \prog{tlmgr} 的程序: `\TL\ 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。

\end{itemize*}

\subsection{获得帮助}
\label{sec:help}

\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 \TL{}，改为购买商业 \TeX{} 系统，在
\url{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:

\begin{description}
\item [起步] 如果你刚刚接触 \TeX，\url{http://tug.org/begin.html}
这个网页提供了这个系统的简短介绍。

\item [\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，它在 \TL{} 的

\OnCD{texmf-dist/doc/generic/FAQ-en/}，也可以在
\url{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。

\item [\TeX{} Catalogue] 如果你在寻找某个特定的宏包、字体、程序等等，\TeX{}
Catalogue 就是你首先该找的地方。这里是所有 \TeX{} 相关内容的一个巨大集合。参见
\url{http://mirror.ctan.org/help/Catalogue/}。

\item [\TeX{} 网上资源] \url{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。

\item [支持信息的归档] 最重要的技术支持论坛是 Usenet 的新闻组
\url{news:comp.text.tex}，邮件列表 \email{texhax@tug.org} 和协作的问题编辑与
回答网站 \url{http://tex.stackexchange.com}。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 \\
\url{http://groups.google.com/groups?group=comp.text.tex} 和 \\
\url{http://tug.org/mail-archives/texhax} 进行查询。当然，一般性的搜索
方式，比如用 \url{http://google.com} 找找，总没有坏处。

\item [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
\dirname{comp.text.tex} 上提问，或者发送邮件到 \email{texhax@tug.org}。
不过，在提问之前\emph{请一定}先阅读 FAQ 上的这一条:
\url{http://www.tex.ac.uk/cgi-bin/texfaq2html?label=askquestion},
它能提高你获得回答的可能性。
Also worth mentioning are the \LaTeX{} Community site at
另外值得提到的有位于 \url{http://www.latex-community.org/} 的
\LaTeX{} Community 和它的论坛
\url{http://www.latex-community.org/forum/} 以及
在 \url{http://tex.stackexchange.com/} 的 \TeX\ StackExchange。

\item [\TL{} 技术支持] 如果你需要报告 bug，或者提出对 \TL{} 的发行、安装或
文档的建议和意见，可以使用 \email{tex-live@tug.org} 这个邮件列表。不过，如果问题
是针对 \TL{} 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 \code{-{}-help} 参数来运行程序都一般能得到用来报告其 bug 的地址。

\end{description}

另一方面，你也不妨帮助其他有问题的朋友，\dirname{comp.text.tex} 与
\code{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。

% don't use \TL so the \uppercase in the headline works. Also so
% tex4ht ends up with the right TeX. Likewise the \protect's.
\section{\protect\TeX\protect\ Live 概览}
\label{sec:overview-tl}

这个小节描述的是 \TL{} 的内容，以及包含 \TL{} 的 \TK{}。

\subsection{\TL, pro\TeX{}t, Mac\TeX\ 的大集合: \TK}
\label{sec:tl-coll-dists}

\TK{} 的 \DVD{} 包含了以下内容:

\begin{description}

\item [\TL] 是一个完整的 \TeX{} 系统，它可以安装在本
地磁盘上。主页: \url{http://tug.org/texlive/}。

\item [Mac\TeX] 在 \TL{} 的基础上增加了原生的 \MacOSX\ 的安装程序和一些其它的
Mac 应用程序。它的主页在 \url{http://tug.org/mactex/}。

\item [pro\TeX{}t] 是 Windows 下的 \MIKTEX\ 发行版的一个增强版本。\ProTeXt\ 在
\MIKTEX\ 基础上增加了一些工具，简化了安装。它完全独立于 \TL{}，有其
自己的安装步骤。主页: \url{http://tug.org/protext}。

\item [CTAN] 一份 \CTAN{} 仓库的快照 (\url{http://www.ctan.org/})。

\end{description}

\CTAN{} 和 \pkgname{protext} 并不一定遵循 \TL{} 的版权协议，因此在分发或修改时要
格外地小心。

\subsection{\TL{} 的顶层目录}
\label{sec:tld}

这里是 \TL{} 发行版顶层目录的一个简短的列表和描述。

\begin{ttdescription}
\item[bin] \TeX{} 系统程序，按平台组织。
%
\item[readme.html] 网页，提供了多种语言的简介和有用的链接。
\item[readme-*.dir] \TL{} 多种语言的简介和有用的链接，同时有 \HTML{}
和纯文本版本。

%
\item[source] 所有程序的源代码，包括主要的基于 \Webc{} 的 \TeX{}
  发行版。
%
\item[texmf-dist] 最主要的文件树，见下文的 \dirname{TEXMFDIST}。
%
\item[tlpkg] 用来维护安装程序所用到的脚本，程序和数据，以及对
Windows 的特殊支持。
\end{ttdescription}

上述目录之外，安装脚本和（多种语言的）\filename{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 \OnCD{doc.html} 文件中提供的完整的链接会有帮助。
几乎所有内容的文档（宏包、格式文件、字体、程序手册，man page，Info 文件等）在
\dirname{texmf-dist/doc} 目录下，因为这些程序本身是属于 \dirname{texmf} 目录的。
\TeX\ 宏包与格式文件的文档则放在 \dirname{texmf-dist/doc} 目录。但不管放在哪个地
方，你都可以使用 \cmdname{texdoc} 程序来寻找这些文档。

\TL\ 本身的文档在 \dirname{texmf-dist/doc/texlive} 目录下，有以下这些语言的版本:

\begin{itemize*}
\item{捷克/斯洛伐克语:} \OnCD{texmf-dist/doc/texlive/texlive-cz}
\item{德语:} \OnCD{texmf-dist/doc/texlive/texlive-de}
\item{英语:} \OnCD{texmf-dist/doc/texlive/texlive-en}
\item{法语:} \OnCD{texmf-dist/doc/texlive/texlive-fr}
\item{意大利语:} \OnCD{texmf-dist/doc/texlive/texlive-it}
\item{波兰语:} \OnCD{texmf-dist/doc/texlive/texlive-pl}
\item{俄语:} \OnCD{texmf-dist/doc/texlive/texlive-ru}
\item{塞尔维亚语:} \OnCD{texmf-dist/doc/texlive/texlive-sr}
\item{简体中文:} \OnCD{texmf-dist/doc/texlive/texlive-zh-cn}

```
\end{itemize*}


\subsection{预定义的 texmf 目录树概览}
\label{sec:texmftrees}

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 \TL{} 的默认布局。\texttt{tlmgr~conf} 命令可
以列出这些变量的值，这样你可以很容易找到它们和你所安装到的目录
名称的对应关系。

所有这些目录树，包括个人的，都应该遵循 \TeX\ 目录结构 (\TDS,
\url{http://tug.org/tds})，包括其中所有细小的子目录，否则文件就
可能找不到。第 \ref{sec:local-personal-macros} 节 (第
\pageref{sec:local-personal-macros} 页) 有更详细的介绍。

\begin{ttdescription}
\item [TEXMFDIST] 这个目录树包含几乎所有原有发行版本的文件——配置文件、脚本、宏
包、字体等等。唯一的例外是每个平台的可执行文件，存储在与它同级的 \code{bin/} 目
录下。
\item [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。
\item [TEXMFHOME] 给用户存放它们自己独立安装的的宏包、字体等等。
这个变量根据不同的用户选择不同的主目录。
\item [TEXMFCONFIG] 给 \verb+texconfig+、\verb+updmap+、和 \verb+fmtutil+
这些程序存储个人修改过的配置文件。
\item [TEXMFSYSCONFIG] 给 \verb+texconfig-sys+、\verb+updmap-sys+ 和
\verb+fmtutil-sys+ 这些程序存储修改过的全局文件.
\item [TEXMFVAR] 这个目录是给 \verb+texconfig+、\verb+updmap+ 和
  \verb+fmtutil+ 存储 (缓存) 格式文件、生成 map 文件这类运行时个人数据的。
\item [TEXMFSYSVAR] 给 \verb+texconfig-sys+、\verb+updmap-sys+ 和
\verb+fmtutil-sys+ 还有 \verb+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，对整个系统都有效。
\item [TEXMFCACHE] \ConTeXt\ MkIV 和 Lua\LaTeX\ 用来保存 (缓存的) 运行时
数据的目录树；缺省为 \code{TEXMFSYSVAR}，如果该目录不可写，则使用
\code{TEXMFVAR}。
\end{ttdescription}

\noindent
默认的目录结构:
\begin{description}
  \item[全系统根目录] 可以包含多个 \TL{} 版本:
  \begin{ttdescription}
    \item[2015] 上一个版本。
    \item[2016] 当前版本。
    \begin{ttdescription}
      \item [bin] ~
      \begin{ttdescription}
        \item [i386-linux] \GNU/Linux 二进制文件
        \item [...]
        \item [universal-darwin] \MacOSX\ 二进制文件
        \item [win32] Windows 二进制文件
      \end{ttdescription}
      \item [texmf-dist\ \ ]      \envname{TEXMFDIST} 和 \envname{TEXMFMAIN}
      \item [texmf-var \ \ ]      \envname{TEXMFSYSVAR}, \envname{TEXMFCACHE}
      \item [texmf-config]        \envname{TEXMFSYSCONFIG}
    \end{ttdescription}
```

```
    \item [texmf-local] \envname{TEXMFLOCAL} 用来存放在不同版本间共享的数据。
  \end{ttdescription}
  \item[用户主 (home) 目录] (\texttt{\$HOME} 或
      \texttt{\%USERPROFILE\%})
    \begin{ttdescription}
      \item[.texlive2015] 给上个版本的，个人生成和配置的数据。
      \item[.texlive2016] 给这个版本的，个人生成和配置的数据。
      \begin{ttdescription}
        \item [texmf-var\ \ \ ] \envname{TEXMFVAR}, \envname{TEXMFCACHE}
        \item [texmf-config]    \envname{TEXMFCONFIG}
      \end{ttdescription}
    \item[texmf] \envname{TEXMFHOME} 个人的宏包文件，等等。
      \textit{等等。}
  \end{ttdescription}
\end{description}


\subsection{\protect\TeX\ 的扩展版本}
\label{sec:tex-extensions}

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 \TL\ 中仍然作为 \prog{tex} 程序出现，在
可见的未来也仍然如此。\TL{} 包括了一些建立在 \TeX\ 基础上的
扩展程序 (也称为 \TeX\ 引擎):

\begin{description}

\item [\eTeX] 为 \TeX\ 增加了一套新的原语 (primitive)。
\label{text:etex} (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 \TeXXeT{} 扩展模式。在默认模式下，\eTeX{} 是与原始的 \TeX\
100\% 兼容的。参见 \OnCD{texmf-dist/doc/etex/base/etex_man.pdf}。

\item [pdf\TeX] 在 \eTeX\ 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 \prog{etex}, \prog{latex} 或 \prog{pdflatex} 这些格式使用的缺省程序。
它的主页在 \url{http://www.pdftex.org/}，文档在
\OnCD{texmf-dist/doc/pdftex/manual/pdftex-a.pdf}。可以在
\OnCD{texmf-dist/doc/pdftex/manual/samplepdf/samplepdf.tex} 找到展示部分功能的例
子。

\item  [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 (\url{http://www.lua.org}) 为
许多棘手的 \TeX{} 问题提供了优雅的解决方案。
当以 \filename{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 \TL\ 中也被作为 Lua 程序的解释器。
见 \url{http://www.luatex.org} 和 \OnCD{texmf-dist/doc/luatex/luatex.pdf}。

\item [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 \url{http://tug.org/xetex}。

\item [\OMEGA\ (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 `\OMEGA{} 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 \TL{}
中了; 改为只支持 Aleph:

\item [Aleph] 将 \OMEGA\ 与 \eTeX\ 扩展合并到一起得到的。
```

参见 \OnCD{texmf-dist/doc/aleph/base}。

\end{description}

\subsection{\protect\TL\ 中其他值得一提的程序}

这里是在 \TL{} 中其他的一些常用程序:

\begin{cmddescription}

\item [bibtex] 参考文献支持。

\item [makeindex, xindy] 索引支持。

\item [dvips] 将 \dvi{} 转换为 \PS{}。

\item [xdvi] X Window System 下的 \dvi{} 阅读器。

\item [dviconcat, dviselect] 从 \dvi{} 文件中复制和粘贴页面。

\item [dvipdfmx] 将 \dvi{} 转换为 PDF，是 (前面提到过的)
pdf\TeX\ 的一套替换方案。

\item [psselect, psnup, \ldots] \PS{} 实用程序。

\item [pdfjam, pdfjoin, \ldots] PDF 实用程序。

\item [context, mtxrun] Con\TeX{}t 和 PDF 处理工具。

\item [htlatex, \ldots] \cmdname{tex4ht}: \AllTeX{} 到 HTML (还有 XML
等其他格式) 的转换器。

\end{cmddescription}

\htmlanchor{installation}
\section{安装}
\label{sec:install}

\subsection{启动安装程序}
\label{sec:inst-start}

首先请找来一张 \TK{} 的 \DVD{}，或者下载 \TL{} 的网络安装程序。参见
\url{http://tug.org/texlive/acquire.html} 以了解更多关于获得这个软件的信息和方法。

\begin{description}
\item [网络安装程序，.zip 或 .tar.gz 格式:] 在 \CTAN 的 \dirname{systems/texlive/tlnet}
目录下可以下载，\url{http://mirror.ctan.org/systems/texlive/tlnet} 这个地址应该
能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 \filename{install-tl.zip} 或者小得多、但只支持 Unix 的
\filename{install-unx.tar.gz}。解压后，\filename{install-tl} 和
\filename{install-tl-windows.bat} 就会出现在 \dirname{install-tl} 子目录中。

\item[网络安装程序，Windows .exe:] 和上述一样从 \CTAN{} 下载，然后双击。它会作为
第一阶段的安装程序和解压工具启动; 见图~\ref{fig:nsis}。它提供了三个选项:
``Simple install'' 启动安装程序向导，``Custom install'' 是给专家提供的 \GUI{} 安
装程序，如第~\ref{sec:wininst}~节所述。第三个选项则仅仅是解压。

\item [\TeX{} Collection \DVD:] 打开 \DVD\ 的 \dirname{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
\url{http://tug.org/usergroups.html}) 或是单独购买 (通过
\url{http://tug.org/store})，又或者是自己从 \ISO\ 镜像刻录。
在多数系统里你都可以直接挂载这个 \ISO{}。在从 \DVD\ 或者 \ISO{} 安装后，
如果还希望从 Internet 获得持续的更新，请看第 \ref{sec:dvd-install-net-updates}
节。

\end{description}

\begin{figure}[tb]
\tlpng{nsis_installer}{.6\linewidth}
\caption{第一阶段的 Windows .exe 安装程序}\label{fig:nsis}
\end{figure}

不管从哪里载入，执行的都是同一个安装程序。两者最明显的区别是通过网络安装
得到的当前可用的程序包。这和和 \DVD\ 或者 \ISO\ 镜像在主版本之间不能更新
不同。

如果你需要用代理服务器来下载，Wget 的代理服务器设置可以使用 \filename{~/.wgetrc}
文件或者环境变量来指定
(\url{http://www.gnu.org/software/wget/manual/html_node/Proxies.html})。
\TL{} 总是使用 \GNU\ Wget 来下载。当然，如果你是从 \DVD\ 或者 \ISO\ 镜像来安装就
没有关系了。

\noindent
下面的章节介绍更详细地介绍了安装程序的启动。

\subsubsection{Unix}

\filename{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:

\begin{alltt}
> \Ucom{perl /path/to/installer/install-tl}
\end{alltt}
(你也许可以直接运行 \Ucom{/path/to/installer/install-tl} 如果这个文件有可执行属
性，或者先 \texttt{cd} 到这个目录中，等等。我们不会把所有这些执行方法列出来。)
你可能需要扩大终端窗口的大小才能在一屏内显示完整的文本安装程序界面
(图~\ref{fig:text-main})。

要在专家 \GUI\ 模式下安装 (见图~\ref{fig:gui-main})，你需要加入了
XFT 支持的 \dirname{Perl::TK}模块，\GNU/Linux 下通常都是这样，
但其他系统下可能不是。这种情况下，你可以运行:
\begin{alltt}
> \Ucom{perl install-tl -gui}
\end{alltt}

要列出所有这些选项:
\begin{alltt}
> \Ucom{perl install-tl -help}
\end{alltt}

\textbf{关于 Unix 下权限的警告: } 在安装过程中，\TL{} 安装程序将会
遵照你的 \code{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 \code{umask 002}。更多关于

\code{umask} 的信息请参见你自己系统的文档。

\textbf{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 \TL{} 安装程序所必须的程序，见第~\ref{sec:cygwin}~节。

\subsubsection{\MacOSX}
\label{sec:macosx}

如第~\ref{sec:tl-coll-dists}~节提到的，我们给 \MacOSX 准备了一套独立的
发行版，叫做 Mac\TeX\ (\url{http://tug.org/mactex})。我们推荐使用原生的
Mac\TeX\ 安装程序，而不是 \TL\ 自带的那个，因为原生的安装程序做了一些
针对 Mac 的调整，尤其是使用 \TeX{}Dist 数据结构方便在 \MacOSX 下的多个 \TeX\ 发
行版 (Mac\TeX, Fink, MacPorts, \ldots) 之间切换。

Mac\TeX\ 是严格依赖 \TL 构建的，所以主 \TeX\ 树和二进制文件也是完全一致的。不过
它添加了一些用来存放 Mac 专有文档和程序的目录。

\subsubsection{Windows}\label{sec:wininst}

如果你使用的是下载 zip 解压后的目录，又或者 \DVD 安装程序无法自动启动了，请
双击 \filename{install-tl-windows.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 \filename{install-tl-advanced.bat}。

你也可以从命令行提示符下启动安装程序。下面 \texttt{>} 表示的就是提示符，
用户输入用 \Ucom{\texttt{bold}} 体表示。如果你正在安装程序目录下，只需要
运行:
\begin{alltt}
> \Ucom{install-tl-windows}
\end{alltt}

或者你也可以通过绝对路径来运行，比如:
\begin{alltt}
> \Ucom{D:\bs{}texlive\bs{}install-tl-windows}
\end{alltt}
这是对 \TK\ \DVD 而言的，假定 \dirname{D:} 是光驱。图~\ref{fig:wizard-w32}
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:
\begin{alltt}
> \Ucom{install-tl-windows -no-gui}
\end{alltt}

要列出所有可用的选项:
\begin{alltt}
> \Ucom{install-tl-windows -help}
\end{alltt}

\begin{figure}[tb]
\begin{boxedverbatim}
Installing TeX Live 2016 from: ...
Platform: i386-linux => 'GNU/Linux on Intel x86'
Distribution: inst (compressed)
Directory for temporary files: /tmp
...
 Detected platform: GNU/Linux on Intel x86

 <B> platforms: 1 out of 17

```
 <S> Set installation scheme (scheme-full)

 <C> customizing installation collections
     47 collections out of 48, disk space required: 4268 MB

 <D> directories:
   TEXDIR (the main TeX directory):
     /usr/local/texlive/2016
   ...

 <O> options:
   ...

 <V> set up for portable installation

Actions:
 <I> start installation to hard disk
 <H> help
 <Q> quit
\end{boxedverbatim}
\caption{文本安装程序主界面 (\GNU/Linux)}\label{fig:text-main}
\end{figure}

\begin{figure}[tb]
\tlpng{install-lnx-main}{\linewidth}
\caption{专家模式 \GUI{} 安装程序界面 (\GNU/Linux)}\label{fig:gui-main}
\end{figure}

\begin{figure}[tb]
\tlpng{wizard-w32}{\linewidth}
\caption{向导模式安装程序界面 (Windows)}\label{fig:wizard-w32}
\end{figure}


\htmlanchor{cygwin}
\subsubsection{Cygwin}
\label{sec:cygwin}

在开始安装之前，请先使用 Cygwin 的 \filename{setup.exe} 程序安装
\filename{perl} 和 \filename{wget} 软件包，如果你还没装过。此外
还推荐安装下列软件包:
\begin{itemize*}
\item \filename{fontconfig} [\XeTeX\ 和 Lua\TeX\ 需要]
\item \filename{ghostscript} [各种实用工具需要]
\item \filename{libXaw7} [\code{xdvi} 需要]
\item \filename{ncurses} [给安装程序提供 \code{clear} 命令]
\end{itemize*}

\subsubsection{文本界面安装程序}

图~\ref{fig:text-main} 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。

\subsubsection{专家图形界面安装程序}

图~\ref{fig:gui-main} 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，这个和文本模式的安装程序没什么区别。

这个模式可以通过下面的命令手工启动:
\begin{alltt}
> \Ucom{install-tl -gui=perltk}
\end{alltt}

\subsubsection{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 (图~\ref{fig:wizard-w32})。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。

在其他平台下，这个模式可以通过下面的命令手工启动:
\begin{alltt}
> \Ucom{install-tl -gui=wizard}
\end{alltt}

\subsection{执行安装程序}
\label{sec:runinstall}

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。

\subsubsection{二进制系统菜单 (只对 Unix 适用)}
\label{sec:binary}

\begin{figure}[tb]
\begin{boxedverbatim}
Available platforms:
==
   a [ ] Cygwin on Intel x86 (i386-cygwin)
   b [ ] Cygwin on x86_64 (x86_64-cygwin)
   c [ ] MacOSX/Darwin universal binaries (universal-darwin)
   d [ ] MacOSX/Darwin on x86_64 (x86_64-darwin)
   e [ ] FreeBSD on x86_64 (amd64-freebsd)
   f [ ] FreeBSD on Intel x86 (i386-freebsd)
   g [ ] GNU/Linux on ARM (armel-linux)
   h [ ] GNU/Linux on ARMhf (armhf-linux)
   i [X] GNU/Linux on Intel x86 (i386-linux)
   j [ ] GNU/Linux on PowerPC (powerpc-linux)
   k [ ] GNU/Linux on x86_64 (x86_64-linux)
   l [ ] NetBSD on x86_64 (amd64-netbsd)
   m [ ] NetBSD on Intel x86 (i386-netbsd)
   o [ ] Solaris on Intel x86 (i386-solaris)
   p [ ] Solaris on Sparc (sparc-solaris)
   s [ ] Solaris on x86_64 (x86_64-solaris)
   t [ ] Windows (win32)
\end{boxedverbatim}
\caption{Binaries (二进制程序) 菜单}\label{fig:bin-text}
\end{figure}

图~\ref{fig:bin-text} 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平台的二进制程序。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。

\subsubsection{选择要安装的组件}
\label{sec:components}

\begin{figure}[tbh]
\begin{boxedverbatim}
Select scheme:
==
 a [X] full scheme (everything)
 b [ ] medium scheme (small + more packages and languages)
 c [ ] small scheme (basic + xetex, metapost, a few languages)
 d [ ] basic scheme (plain and latex)
 e [ ] minimal scheme (plain only)
 f [ ] ConTeXt scheme
 g [ ] GUST TeX Live scheme
 h [ ] teTeX scheme (more than medium, but nowhere near full)
 i [ ] XML scheme
 j [ ] custom selection of collections
\end{boxedverbatim}
\caption{Scheme (安装方案) 菜单}\label{fig:scheme-text}
\end{figure}

图~\ref{fig:scheme-text} 展示了 \TL\ 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 \optname{full}
方案会把所有可用的都装上。这是推荐方案，不过你也可以给小点的系统选择
\optname{basic} 方案，为测试选用 \optname{minimal} 方案，又或者是
介乎其间的 \optname{medium} 或 \optname{teTeX} 方案。还有许多特殊或者
专门针对特定国家的方案。

\begin{figure}[tb]
\tlpng{stdcoll}{.7\linewidth}
\caption{Collections (集合) 菜单}\label{fig:collections-gui}
\end{figure}

你可以使用 `collections' 菜单来详细
选择安装方案。(图~\ref{fig:collections-gui} 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 \Dash 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
(\TL 中最小的组织单位) 则包含了实际的 \TeX\ 宏文件，字体文件，等等。

如果你觉得 collection 菜单对安装控制还不够细，可以在安装后使用
\TeX\ Live Manager (\prog{tlmgr}) 程序 (参见第~\ref{sec:tlmgr}~节),
它能在软件包一层控制安装。

\subsubsection{目录}
\label{sec:directories}

缺省的目录布局在第~\ref{sec:texmftrees}~节有过叙述，见
第~\pageref{sec:texmftrees}~页。在 Unix 下默认的安装目录是
\dirname{/usr/local/texlive/2016} 而 Windows 下是
|%SystemDrive%\texlive\2016|。这样的安排允许你有多个并行存在的 \TL\ 安装，每年的
发行版本一个，你可以通过修改搜索路径来在它们中间切换。

这个安装路径可以通过设置安装程序中的 \dirname{TEXDIR} 来修改。这个选项和其他选项
如图~\ref{fig:gui-main} 所示。最常见的更改它的原因要么是该分区没有足够的空间（完
整的 \TL\ 安装需要好几 GB 的空间），要么是没有默认位置的写权限。（虽然要安装
\TL\ 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。）

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

一个合理的选择是你自己主目录下的一个子目录，尤其在只有你一个人
使用的时候。使用 `|~|' 来表示主目录，比如 `|~/texlive/2016|'。

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
\TL{}。（你可能希望还维护一个类似 \dirname{/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无
关了，你只要在新版本出来直接重新指向符号链接即可。）

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

\dirname{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 \dirname{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
\dirname{$HOME}，而 Windows 下展开为 \verb|%USERPROFILE%|。
多加一句：\envname{TEXMFHOME} 和所有的目录树一样，都必须按照
\TDS\ 组织，否则文件可能会找不到。

\dirname{TEXMFVAR} 是用来给每个用户存储大多数运行时缓存数据
的。\dirname{TEXMFCACHE} 则是给 Lua\LaTeX\ 和 \ConTeXt\ MkIV 存储缓存
（见第 \pageref{sec:context-mkiv} 页的 \ref{sec:context-mkiv} 节）；它的默认值是
\dirname{TEXMFSYSVAR}，或者如果 \dirname{TEXMFVAR}，如果这个目录可以写。

\subsubsection{选项}
\label{sec:options}

\begin{figure}[tbh]
\begin{boxedverbatim}
Current setup:
 <P> use letter size instead of A4 by default: [ ]
 <E> execution of restricted list of programs: [X]
 <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:
 <Y> after installation, get package updates from CTAN: [X]
\end{boxedverbatim}
\caption{Options 菜单 (Unix)}\label{fig:options-text}
\end{figure}

图~\ref{fig:options-text} 显示了文本模式的选项菜单。关于这个菜单的
每个选项：

\begin{description}
\item[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。

\item[execution of restricted list of programs:] 在 \TL\ 2010 中，
默认允许执行一些外部程序。这（很少的一部分）允许的程序列表在
\filename{texmf.cnf} 中定义。参见 2010 版新闻（第~\ref{sec:2010news} 节）
以了解更多信息。

\item[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用户个人的 \dirname{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。

\item[install font/macro \ldots\ tree:] 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。

\item[create symlinks in standard directories:]
这个选项（只对 Unix 有效的）可以省下设定环境变量的步骤。如果
没有选择它，就必须把 \TL{} 的对应目录添加到 \envname{PATH},
\envname{MANPATH} 和 \envname{INFOPATH} 中。如果要创建符号链
接，你需要对这些目标目录的写权限。我们强烈建议不要\emph{不要}
用这个命令来覆盖现有的 \TeX\ 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 \TeX\ 文件。

\item[after installation \ldots\ \CTAN:] 从 \DVD\ 安装时，这个选项
被缺省启用，因为通常你会希望在此后通过 \CTAN\ 安装一年内更新的软件
包。禁用它唯一可能的原因是你只安装了 \DVD\ 内容的一部分，计划在以后
扩展安装。无论如何，安装程序时和安装后的更新的软件包仓库如果需要可
以分别设置，见第~\ref{sec:location}~节和
第~\ref{sec:dvd-install-net-updates}~节。
\end{description}

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~\ref{sec:postinstall}~节来了解还需要做些什么工作。

\subsection{install-tl 命令行选项}
\label{sec:cmdline}

输入
\begin{alltt}
> \Ucom{install-tl -help}
\end{alltt}
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:

\begin{ttdescription}
\item[-gui] 尽可能用 \GUI{} 模式的安装程序。它需要安装了
  编译进 XFT 支持的 Perl/Tk 模块
  (\url{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。

\item[-no-gui] 强制使用文本模式安装程序，就算在 Windows 下也是如此。

\item[-lang {\sl LL}] 指定安装程序界面的语言，使用标准的（通常是两个
两个字符）语言代码。安装程序会尝试自己判断出合适的语言，如果判断出的
语言没有支持就会使用英语替代。可以运行 \verb|install-tl --help| 获得
支持的语言列表。

\item[-portable] 为比如 \USB{} 盘上便携使用安装。也可以在
文本安装程序中用 \code{V} 命令选择，或者 GUI 安装程序中选择，
参见第 \ref{sec:portable-tl} 节。

\item[-profile {\sl profile}] 载入安装配置文件 \var{file} 以不需要
用户干预的方式完成安装。安装程序会在安装到的 \dirname{tlpkg}
子目录中创建一个叫 \filename{texlive.profile} 的文件。这个文件可以
用作参数，在不同的系统下完成完全一致的安装。或者你也可以使用一个定
制的配置文件，一般通过裁剪生成的文件得到，或者空文件，这样安装时会
使用所有缺省值。

\item [-repository {\sl url-or-directory}] 指定作为来源的软件包仓库，
参见下文。

\htmlanchor{opt-in-place}
\item[-in-place] （补充说明：除非你清楚自己要做什么，否则不要使用这个选项。）
如果你已经有一份用 rsync，svn，或者其他方式安装的
\TL{}（参见 \url{http://tug.org/texlive/acquire-mirror.html}）那
这个选项会使用现有的，只执行必要的安装后操作。注意 \filename{tlpkg/texlive.tlpdb}
文件会被覆盖，你需要自己备份它。另外，文件的删除必须手动完成。所以
只在你知道自己在干什么的情况下才用它。这个选项不能在安装程序界面中
选择。

\end{ttdescription}

\subsubsection{\optname{-repository} 参数}
\label{sec:location}

默认的网络软件包安装位置是由 \url{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 \texttt{ftp:}，\texttt{http:} 或
\texttt{file:/} 起始的 URL，或者本地路径。（如果给定了一个
\texttt{http:} 或 \texttt{ftp:} 地址，其末尾的 `\texttt{/}' 字符
和末尾的 `\texttt{/tlpkg}' 这段都会被忽略。）

比如你可以选择这样的一个 \CTAN\ 镜像：
\url{http://ctan.example.org/tex-archive/systems/texlive/tlnet/}，
当然你应该把 |ctan.example.org/tex-archive| 替换为具体镜像的
域名和特定的顶层 \CTAN\ 路径（\url{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 \texttt{file:/}
开头的 URL），就会使用指定路径下 \dirname{archive} 子目录中的
压缩文件进行安装。

\subsection{安装后的操作}
\label{sec:postinstall}

安装后可能需要一些额外的操作。

\subsubsection{Unix 下的环境变量}

\label{sec:env}

如果你选择了在标准路径下创建符号连接 (在第~\ref{sec:options}~节提到),
那就不需要设置环境变量了。否则，在 Unix 系统中，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。(在 Windows 下安装程序会负责这一步。)

每个支持的平台都在 \dirname{TEXDIR/bin} 下有自己的子目录。
参见图~\ref{fig:bin-text} 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 \envname{PATH} 后，手册页页可以被自动
找到。

对于 \prog{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 \filename{$HOME/.profile}
(或者其他由 \filename{.profile} 载入的文件)，应该添加的内容应该类似
这样:

```
\begin{sverbatim}
PATH=/usr/local/texlive/2016/bin/i386-linux:$PATH; export PATH
MANPATH=/usr/local/texlive/2016/texmf-dist/doc/man:$MANPATH; export MANPATH
INFOPATH=/usr/local/texlive/2016/texmf-dist/doc/info:$INFOPATH; export INFOPATH
\end{sverbatim}
```

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

```
\begin{sverbatim}
setenv PATH /usr/local/texlive/2016/bin/i386-linux:$PATH
setenv MANPATH /usr/local/texlive/2016/texmf-dist/doc/man:$MANPATH
setenv INFOPATH /usr/local/texlive/2016/texmf-dist/doc/info:$INFOPATH
\end{sverbatim}
```

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 \TL\ 的这些目录加进去就行了。

\subsubsection{环境变量的全局配置}
\label{sec:envglobal}

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 \filename{/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:

```
\begin{sverbatim}
MANPATH_MAP /usr/local/texlive/2016/bin/i386-linux \
            /usr/local/texlive/2016/texmf-dist/doc/man
\end{sverbatim}
```

然后 2)~检查 \filename{/etc/environment} 是否定义了默认的搜索路径
和其他的默认环境变量。

在每个 (Unix) 二进制目录下，我们都会创建一个 \code{man} 符号链接
到 \dirname{texmf-dist/doc/man} 目录。因为
有些 \code{man} 程序，比如 \MacOSX\ 标准的 \code{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。

\subsubsection{\DVD\ 安装后的网络更新}
\label{sec:dvd-install-net-updates}

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

\begin{alltt}
> \Ucom{tlmgr option repository http://mirror.ctan.org/systems/texlive/tlnet}
\end{alltt}

这告诉 \cmdname{tlmgr} 从就近的 \CTAN\ 镜像获取未来更新。
从 \DVD{} 安装时会默认完成这一步，通过第~\ref{sec:options}~节介绍
的选项。

如果自动镜像选择出现了问题，你可以从 \url{http://ctan.org/mirrors}
列表中自己指定一个 \CTAN\ 镜像。使用与上述一致的 \dirname{tlnet}
子目录路径。

\subsubsection{\XeTeX{} 和 Lua\TeX{} 的系统字体配置}
\label{sec:font-conf-sys}

\XeTeX\ 和 Lua\TeX\ 可以使用任何系统安装的字体，而不只是 \TeX\ 目录树
中的那些。它们使用类似但不完全一致的方式实现这一功能。

在 Windows 和 \MacOSX\ 下 \TL\ 提供的字体会自动为 \XeTeX\ 所用。
但如果你在其他 Unix 系统中安装了 \filename{xetex} 软件包，则需要把系
统配置一番 \XeTeX\ 才能找到随 \TL\ 安装的那些字体。

为了进行配置，\pkgname{xetex} 安装后 (不管是初始安装还是后来安装的) 都会在
\filename{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 \TL{} 的字体 (假定你有足够的权限)，请依照下面的
步骤来做:
\begin{enumerate*}
\item 将 \filename{texlive-fontconfig.conf} 文件复制到
\dirname{/etc/fonts/conf.d/09-texlive.conf}。
\item 运行 \Ucom{fc-cache -fsv}。
\end{enumerate*}

如果你没有足够的权限执行上述操作，或者只需要把 \TL{} 字体提供给你自
己，可以这么做:
\begin{enumerate*}
\item 将 \filename{texlive-fontconfig.conf} 文件复制到
      \filename{~/.fonts.conf}，其中 \filename{~} 是你的主目录。
\item 运行 \Ucom{fc-cache -fv}。
\end{enumerate*}

你可以运行 \code{fc-list} 来查看系统字体的名称。命令
\code{fc-list : family style file spacing} 可以列出一些有趣的信息。

\subsubsection{\ConTeXt{} Mark IV}
\label{sec:context-mkiv}

“旧”的 \ConTeXt{} (Mark II) 和“新的” \ConTeXt{} (Mark IV)
应该在 \TL{} 安装后直接就能运行，而且只要你一直用 \verb+tlmgr+ 来更新，

它不应该需要任何其他的处理。

然而，因为 \ConTeXt{} MkIV 没有使用 kpathsea 库，在你手动安装
文件（没使用 \verb+tlmgr+）后必须执行一定的设置。在每次安装后，每个
MkIV 用户必须执行：

\begin{sverbatim}
context --generate
\end{sverbatim}
来刷新 \ConTeXt{} 磁盘缓存数据。
生成的文件会被保存在 \code{TEXMFCACHE} 目录下，在 \TL{} 中这个目录
的默认值是 \verb+TEXMSYSVAR;TEXMFVAR+。

\ConTeXt\ MkIV 会读取 \verb+TEXMFCACHE+ 提到的所有路径，
并写入第一个可写东风路径。在读取时，最后找到的匹配会被优先选择。

要了解更多信息，参见
\url{http://wiki.contextgarden.net/Running_Mark_IV}。

\subsubsection{集成本地与个人宏文件}
\label{sec:local-personal-macros}

这在第~\ref{sec:texmftrees}~节已经顺带提到过了：\dirname{TEXMFLOCAL}
目录（它的默认值是 \dirname{/usr/local/texlive/texmf-local} 或者
Windows 下的 \verb|%SystemDrive%\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 \dirname{TEXMFHOME} 目录（其
默认值是 \dirname{$HOME/texmf} 或者 \verb|%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 \TL{}
版本之间共享，每个版本都能看到其内容。因此不应该把
\dirname{TEXMFLOCAL} 改得和主 \TL{} 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 \TeX\ 目录结构（\TDS）
子目录中，参见 \url{http://tug.org/tds} 或者
\filename{texmf-dist/web2c/texmf.cnf} 文件。比如一个 \LaTeX{} 文档类或者宏包应该
放在 \dirname{TEXMFLOCAL/tex/latex} 或者 \dirname{TEXMFHOME/tex/latex}
目录下，要不然就是它们的一个子目录下。

\dirname{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 \cmdname{mktexlsr} 命令或者
\TeX\ Live Manager \GUI{}, configuration 选项卡中的
`Reinit file database' 按钮来刷新它。

默认情况下，这些变量的每一个都定义为所示单独的目录。这不是一个硬性
规定。如果你需要在某些大的宏包的不同版本直接方便地切换，可以维护多
个自己的目录树，这通过把 \dirname{TEXMFHOME} 设置为目录列表实现，
大括号包围，逗号分隔：

\begin{verbatim}
  TEXMFHOME = {/my/dir1,/mydir2,/a/third/dir}
\end{verbatim}

第~\ref{sec:brace-expansion}~节进一步介绍了括号展开。

\subsubsection{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 \TeX{} 安装的细节，

否则请不要涉足这个领域。\TL\ 已经包含了很多字体，所以请先查查。

\XeTeX\ 或 Lua\TeX\ 是可行的替代方案 (参见第~\ref{sec:tex-extensions}~节)，
它们能让你使用操作系统的字体而不必将它安装到 \TeX 中。

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

\subsection{测试安装是否成功}
\label{sec:test-install}

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

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

这个小节后面给出了一些测试系统是否正常工作的基本步骤。我们这里使用的是 Unix 命令，
在 \MacOSX{} 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。

\begin{enumerate}

\item 首先确认你可以执行 \cmdname{tex} 程序:
\begin{alltt}
> \Ucom{tex -{}-version}
TeX 3.14159265 (TeX Live ...)
Copyright ... D.E. Knuth.
...
\end{alltt}
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 \dirname{bin} 子目录添加
到 \envname{PATH} 中。参见第~\pageref{sec:env}~页关于设置环境变量
的说明。

\item 处理一个基本的 \LaTeX{} 文件:
\begin{alltt}
> \Ucom{latex sample2e.tex}
This is pdfTeX 3.14...
...
Output written on sample2e.dvi (3 pages, 7484 bytes).
Transcript written on sample2e.log.
\end{alltt}
如果无法找到 \filename{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 \TeX 相关的环境变量然后重试。(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~\pageref{sec:debugging}~页的``调试操作''一节以了解更多信息。)

\item 即时预览结果:
\begin{alltt}
> \Ucom{xdvi sample2e.dvi}   # Unix
> \Ucom{dviout sample2e.dvi} # Windows
\end{alltt}
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。) \cmdname{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
\envname{DISPLAY} 环境变量设置错误，都会得到
\samp{Can't open display} 这句错误信息。

\item 创建用于打印或显示的 \PS{} 文件:
\begin{alltt}
> \Ucom{dvips sample2e.dvi -o sample2e.ps}
\end{alltt}

\item 创建 PDF 文件而非 \dvi{}，这里采用直接处理 \filename{.tex}
文件输出 PDF 的方式:
\begin{alltt}
> \Ucom{pdflatex sample2e.tex}
\end{alltt}

\item 预览 PDF 文件:
\begin{alltt}
> \Ucom{gv sample2e.pdf}
\textrm{或: }
> \Ucom{xpdf sample2e.pdf}
\end{alltt}
\cmdname{gv} 和 \cmdname{xpdf} 现在都不包含在 \TL{} 中，你
必须单独安装它们。请分别参阅 \url{http://www.gnu.org/software/gv} 和
\url{http://www.foolabs.com/xpdf}。(还有许多其他的 PDF 查看器。)
Windows 下我们推荐 Sumatra PDF
(\url{http://blog.kowalczyk.info/software/sumatrapdf})。

\item 除 \filename{sample2e.tex} 外可能会对你有用的标准测试文件:

\begin{ttdescription}
\item [small2e.tex] 比 \filename{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。
\item [testpage.tex] 测试你的打印机是否带有预设的偏移量。
\item [nfssfont.tex] 打印一份字体表格 (proofsheet) 以供测试。
\item [testfont.tex] 用 plain \TeX{} 打印字体表格。
\item [story.tex] 最经典的 (plain) \TeX{} 测试文件。在执行
\samp{tex story.tex} 之后，你还要在 \code{*} 提示符下键入 \samp{\bs bye}。
\end{ttdescription}
你可以用与我们处理 \filename{sample2e.tex} 文件时相同的方式来处理这些文件。

\item 如果你安装了 \filename{xetex} 包，可以按如下步骤测试它能否访问系统字体:
\begin{alltt}
> \Ucom{xetex opentype-info.tex}
This is XeTeX, Version 3.14\dots
...
Output written on opentype-info.pdf (1 page).
Transcript written on opentype-info.log.
\end{alltt}

如果你收到 ``Invalid fontname `Latin Modern Roman/ICU'\dots'' 这样
的错误信息，就说明需要配置系统才能找到 \TL{} 自带的字体。
参见第~\ref{sec:font-conf-sys}~节。

\end{enumerate}

\subsection{其他可下载软件的链接}

```
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 \url{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。
\begin{description}
\item[Ghostscript] \url{http://www.cs.wisc.edu/~ghost/}
\item[Perl] \url{http://www.perl.org/} 与 CPAN 中的补充包，
    \url{http://www.cpan.org/}
\item[ImageMagick] \url{http://www.imagemagick.com}，用于图形处理和转换
\item[NetPBM] \url{http://netpbm.sourceforge.net/}，同样用于图形。

\item[面向 \TeX\ 的编辑器] 有很广泛的选择，一般依用户个人的口味而定。这
     里按字典序列出了一些（部分是 Windows 才有的）。
  \begin{itemize*}
  \item \cmdname{GNU Emacs} 在 Windows 下的原生版本在
        \url{http://www.gnu.org/software/emacs/windows/ntemacs.html}。
  \item \cmdname{Emacs 的 Auc\TeX\ 包} 的 Windows 版本在 \CTAN\ 提供。
        AuC\TeX\ 的主页在 \url{http://www.gnu.org/software/auctex}。
  \item \cmdname{LEd} 在 \url{http://mirror.ctan.org/support/LEd} 提供。
  \item \cmdname{SciTE} 在
        \url{http://www.scintilla.org/SciTE.html} 提供。
  \item \cmdname{Texmaker} 是自由软件，在
        \url{http://www.xm1math.net/texmaker} 提供。
  \item \cmdname{TeXstudio} 是 \cmdname{Texmaker} 的一个
  fork，引入了额外的功能；\url{http://texstudio.sourceforge.net}。
  \item \cmdname{TeXnicCenter} 是自由软件，在
        \url{http://www.texniccenter.org} 提供，也随 pro\TeX{}t
        发行版附带。
  \item \cmdname{TeXworks} 是自由软件，在 \url{http://tug.org/texworks} 提供，也
        作为 \TL{} 的一部分只在 Windows 被安装。
  \item \cmdname{Vim} 是自由软件，在
        \url{http://www.vim.org} 提供。
  \item \cmdname{WinEdt} 是共享软件，在
        \url{http://tug.org/winedt} 或 \url{http://www.winedt.com} 提供。
  \item \cmdname{WinShell} 在 \url{http://www.winshell.de} 提供。
  \end{itemize*}
\end{description}
关于这类软件包和程序，\url{http://tug.org/interest.html} 有一份更长的列表。

\section{特殊安装}

前面的章节描述了基本的安装过程。这里我们介绍一些特殊的情形。

\htmlanchor{tlsharedinstall}
\subsection{用户共享（或跨机器）安装}
\label{sec:sharedinstall}

\TL{} 的设计可以使它在同一个系统的不同的用户间共享，或者可以在网络上不
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：\TL{} 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
\filename{$TEXMFDIST/web2c/texmf.cnf} 配置文件中看到实际的处理，
它包含了类似下面的内容：
\begin{verbatim}
TEXMFROOT = $SELFAUTOPARENT
...
TEXMFDIST = $TEXMFROOT/texmf-dist
...
```

```
TEXMFLOCAL = $SELFAUTOPARENT/../texmf-local
\end{verbatim}
```
这就意味着，其它的系统或用户只需要把 \TL{} 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 \TL{} 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 \url{http://tug.org/texlive/w32client.html} 下载到一个叫 \filename{tlaunch} 的启动程序。它的主窗口的菜单和按钮包括许多 \TeX\ 相关功能的程序和文档。

首次使用时，它会修改 \TL\ 的搜索路径并创建一些文件关联。它还会创建一个开始菜单项目来取消这些配置，在这个启动器自身的菜单里也有这个选项。

所有这些配置都是在一个 ini 文件里设置的。你可以自己编辑这个文件来添加比如 SumatraPDF 或者某个 \LaTeX\ 编辑器的信息。参见上述网站获得更多的信息。

```
\htmlanchor{tlportable}
\section{便携 (\USB{}) 安装}
\label{sec:portable-tl}
```

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

要使用这样的便携安装来运行 \TeX{}，你必须将对应的二进制目录加入终端的搜索路径，如往常一样。在 Windows 下，你可以双击安装根目录下的 \filename{tl-tray-menu} 来选择执行一些常见任务，如这个截图所示：

```
\medskip
\tlpng{tray-menu}{4cm}
\smallskip
```

\noindent `More\ldots' 项目解释了如何定制这个菜单。

```
\htmlanchor{tlisoinstall}
\subsection{\ISO\ (或 \DVD) 安装}
\label{sec:isoinstall}
```

如果你不需要经常更新或者修改安装，或者有其他可以使用 \TL{} 的系统，创建一个 \TL{} 安装的 \ISO\ 可能会比较方便，因为：

```
\begin{itemize}
\item 在不同计算机之间复制 \ISO\ 要比复制普通安装快得多。
\item 如果你双启动不同的操作系统而且希望它们共享一个
  \TL{} 的安装，\ISO\ 安装不会受不同文件系统支持
  (FAT32, NTFS, HFS+) 的限制。
\item 虚拟机可以直接挂载这样的 \ISO{}。
\end{itemize}
```

当然你还可以把 \ISO\ 刻录到 \DVD{} 上，如果有用的话。

桌面 \GNU/Linux/Unix 系统，包括 \MacOSX{}，都能够挂载 \ISO{}。Windows 8 是第一个 (!) 支持这个功能的 Windows 版本。此外，没有任何与普通硬盘安装不同的地方，参见第 \ref{sec:env} 节。

在准备这样的 \ISO\ 安装时，最好忽略发布年份的子目录，并让 \filename{texmf-local}
处在和其他目录树（\filename{texmf-dist}, \filename{texmf-var} 等）同级。
你可以在安装程序的普通目录选项中设置这些。

对于一个物理的（而不是虚拟的）Windows 系统，你可以将 \ISO\ 刻录在
DVD 上，不过花点时间在免费的 \ISO\ 挂载工具上是值得的，比如
\url{http://wincdemu.sysprogs.org/} 的 WinCDEmu。

至于 Windows 系统集成，你可以包含第~\ref{sec:sharedinstall} 节描述的
\filename{w32client} 脚本（在 \url{http://tug.org/texlive/w32client.html}），
它在 \ISO\ 里和在网络安装时一样好用。

在 \MacOSX{}，如果符号链接 \filename{/usr/texbin} 指向对应的二进制目录，
TeXShop 会使用 DVD 安装，比如：
\begin{verbatim}
sudo ln -s /Volumes/MyTeXLive/bin/universal-darwin /usr/texbin
\end{verbatim}

历史说明：\TL{} 2010 是第一个不支持“live”发布的 \TL{} 版本，但要在
\DVD\ 或者 \ISO\ 上执行一直是需要一定技巧的，尤其是必须设置一些环境
变量。如果你从现有的安装创建 \ISO\ 就不需要了。

\htmlanchor{tlmgr}
\section{\cmdname{tlmgr}：管理你的安装}
\label{sec:tlmgr}

\begin{figure}[tb]
\tlpng{tlmgr-gui}{.8\linewidth}
\caption{\GUI\ 模式的 \prog{tlmgr}：按下 `Load' 后的主窗口。}
\label{fig:tlmgr-gui}
\end{figure}

\begin{figure}[tb]
\hbox to \hsize{%
  \vtop{\hsize=.59\linewidth
    \null % make figures align at the top
    \tlpng{tlmgr-general-options}{\hsize}
    \caption{\GUI\ 模式的 \texttt{tlmgr}：通用选项}
    \label{fig:tlmgr-general-options}
  }
  \hfil
  \vtop{\hsize=.39\linewidth
    \null
    \tlpng{tlmgr-paper-options}{\hsize}
    \caption{\GUI\ 模式的 \texttt{tlmgr}：纸张尺寸选项}
    \label{fig:tlmgr-paper-options}
  }
}
\end{figure}

\TL{} 包含一个叫 \prog{tlmgr} 的程序，它可以用来管理安装后的
系统。它的功能包括：

\begin{itemize*}
\item 列出方案（scheme），集合和安装包；
\item 安装、升级、备份、恢复、卸载软件包，并且能自动计算依赖关系；
\item 查找和列出软件包以及它们的描述；

```
\item 列出、添加和删除不同平台的可执行文件;
\item 改变安装选项，比如纸张大小和源文件位置 (参见第~\ref{sec:location}~节)。
\end{itemize*}

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

\subsection{\cmdname{tlmgr} 的 \GUI{} 模式}
\prog{tlmgr} 可以用以下命令以图形模式 (图~\ref{fig:tlmgr-gui}) 启动:
\begin{alltt}
> \Ucom{tlmgr -gui}
\end{alltt}
Windows 下可以通过开始菜单: \texttt{开始}, \texttt{程序},
\texttt{TeX Live ...}, \texttt{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的软件包。

图~\ref{fig:tlmgr-general-options} 和~\ref{fig:tlmgr-paper-options}
展示了通用和纸张尺寸选项界面。

\subsection{\cmdname{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:
\begin{alltt}
> \Ucom{tlmgr update -all}
\end{alltt}
如果这太激进了一点，先尝试:
\begin{alltt}
> \Ucom{tlmgr update -all -dry-run}
\end{alltt}
或 (产生更少输出):
\begin{alltt}
> \Ucom{tlmgr update -list}
\end{alltt}

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。

\begin{alltt}
> \Ucom{tlmgr -repository /local/mirror/tlnet install collection-xetex}
\end{alltt}
它会产生下面的输出 (节略部分):
\begin{fverbatim}
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/2016/texmf-dist/ls-R...
...
running fmtutil-sys --missing
...
Transcript written on xelatex.log.
```

```
fmtutil: /usr/local/texlive/2016/texmf-var/web2c/xetex/xelatex.fmt installed.
\end{fverbatim}

如你所见，\prog{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）：
\begin{alltt}
> \Ucom{tlmgr show collection-latexextra}
\end{alltt}
会产生这样的输出：
\begin{fverbatim}
package:    collection-latexextra
category:   Collection
shortdesc:  LaTeX supplementary packages
longdesc:   A very large collection of add-on packages for LaTeX.
installed:  Yes
revision:   32768
\end{fverbatim}

最后也是最重要的，查阅 \url{http://tug.org/texlive/tlmgr.html}
这里的完整文档，或者：
\begin{alltt}
> \Ucom{tlmgr -help}
\end{alltt}

\section{有关 Windows 平台的说明}
\label{sec:windows}

\subsection{针对 Windows 的特征}
\label{sec:winfeatures}

在 Windows 下，安装程序额外地做了以下一些事情：
\begin{description}
\item[菜单与快捷方式。] 在开始菜单上加入了新的`\TL{}'程序菜单，主要是一些
\GUI{} 程序（如 \prog{tlmgr}、\prog{texdoctk}、PS\_View (\prog{psv})，它是
PostScript 预览程序）和一些文档的菜单。
\item[文件关联。] 如果启用，\prog{TeXworks}、\prog{Dviout}
和 \prog{PS\_view} 会成为它们对应文件类型的默认程序，或者在
这些文件类型的“用...打开”右键菜单中出现。
\item[位图到 eps 转换器。] 许多位图格式会在它们的“用...打开”菜单中获得一个
  \cmdname{bitmap2eps} 项目。Bitmap2eps 是一个使用 \cmdname{sam2p} 或
  \cmdname{bmeps} 的简单脚本。
\item[自动路径调整。] 不需要手动配置步骤。
\item[自动设置环境变量。] 不再需要手动的配置步骤。
\item[反安装程序。] 安装程序为 \TL{} 创建了`添加/删除程序'条目。这与
\TeX\ Live Manager \GUI\ 的删除按钮相对应。对于单用户安装，安装程序会在开始菜单
创建卸载项目。
\item[写保护。] 对于管理员安装，\TL\ 的目录会有写保护，至少在 \TL\ 被安装到 NTFS
格式的本地磁盘时是这样的。
\end{description}

\subsection{Windows 上附加的软件}

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

```
\begin{description}
\item[Perl 和 Ghostscript。] 由于 Perl 和 Ghostscript 的重要性，\TL{}
提供了这些程序的`隐藏'拷贝。需要这些软件支持的 \TL{} 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。

\item[PS\_View。] 同时安装的还有 PS\_View，一款 \PS{} 和 PDF
浏览器；见图~\ref{fig:psview}。

\begin{figure}[tb]
\tlpng{psview}{.6\linewidth}
\caption{PS\_View: 可以获得很高的放大倍数！}\label{fig:psview}
\end{figure}

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

\item[\TeX{}works.]  \TeX{}works 是一个集成了 PDF 阅读器的
  \TeX\ 编辑器。

\item[命令行工具。] 与常见的 \TL{} 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 \cmdname{gzip}、
\cmdname{unzip} 以及来自于 \cmdname{xpdf} 套装中的一些工具
(\cmdname{pdfinfo}, \cmdname{pdffonts}, \ldots)。
虽然 \cmdname{xpdf} 浏览器本身没有 Windows 版本，但你可以从
\url{http://blog.kowalczyk.info/software/sumatrapdf}
下载基于 xpdf 的 Sumatra PDF 浏览器，这是一种解决办法。

\item[fc-list、fc-cache, \ldots] 来自于 fontconfig 库的这些工具有助于 \XeTeX{}
处理 Windows 的系统字体。你可以使用 \prog{fc-list} 来确定传递给经 \XeTeX
扩展后的 \cs{font} 命令的字体名称。如果需要，首先运行 \prog{fc-cache} 更新
字体信息。
\end{description}


\subsection{User Profile 目录相当于主目录}
\label{sec:winhome}

Windows 下对应于 Unix 下的主目录的是 \verb|%USERPROFILE%|。
在 Windows XP 下，它通常位于
\verb|C:\Documents and Settings\<username>|;
在 Windows Vista 和之后的版本下是 \verb|C:\Users\<username>|。通常情况下，在
\filename{texmf.cnf} 文件和 \KPS{} 中，\verb|~| 在 Windows 和 Unix 下均可以
进行合适的展开。

\subsection{Windows 注册表}
\label{sec:registry}

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 \path{HKEY_CURRENT_USER}
和 \path{HKEY_LOCAL_MACHINE}，通常简写为 \path{HKCU} 和 \path{HKLM}。注册表的
\path{HKCU} 部分位于用户的主目录（参见~\ref{sec:winhome}~节）。\path{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。

\subsection{Windows 权限}
\label{sec:winpermissions}

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。我们努力使得 \TL{} 的安装不需要管理员权限。

如果安装程序以管理员权限启动，会有选项允许给所有用户安装，如果启用了这个选项，就会给所有用户创建快捷方式，并修改系统环境变量。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，\TL{} 预设的默认根目录总是位于 \verb|%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 \TL{} 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 \TL{} 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 \TL{}。如果安装了 \TeX{}works，其快捷方式也将 \TL{} 加进了搜索路径中，所以它应该不会出现路径问题。

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

\subsection{增加 Windows 和 Cygwin 下的最大内存量}
\label{sec:cygwin-maxmem}

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

对 Cygwin，你可以按照 Cygwin 用户指南 (\url{http://www.cygwin.com/cygwin-ug-net/setup-maxmem.html}) 中的办法增加可用内存量。

对 Windows，你必须创建一个文件，例如命名为 \code{morememem.reg}，包含如下内容：

\begin{sverbatim}
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\Software\Cygwin]
"heap_chunk_in_mb"=dword:ffffff00
\end{sverbatim}

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

\section{Web2C 用户指南}

\Webc{} 是一整套 \TeX{} 相关程序的集合：\TeX{} 本身、\MF{}、\MP{}、\BibTeX{}，等等。它是 \TL{} 的核心。\Webc{} 的主页及最新的手册等都在 \url{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。

\Webc{} 系统可以在 Unix、32 位 Windows 系统、\MacOSX{} 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的 \TeX{} 程序包括：

\begin{cmddescription}
\item[bibtex]   维护参考文献。
\item[dvicopy]   展开 \dvi{} 中的虚拟字体 (virtual font) 引用。
\item[dvitomp]   将 \dvi{} 转换为 MPX (MetaPost 图片)。
\item[dvitype]   将 \dvi{} 转换为可读文本。
\item[gftodvi]   生成 Generic 格式字体的 proofsheet。
\item[gftopk]   将 Generic 格式字体转换为 packed 格式字体。
\item[gftype]   将 Generic 格式字体转换为可读文本。
\item[mf]   创建字体。
\item[mft]   以漂亮的方式排版输出 \MF{} 的代码。
\item[mpost]   创建技术性插图。
\item[patgen]   创建断字规则文件。
\item[pktogf]   将 Packed 格式字体转换为 generic 格式字体。
\item[pktype]   将 PK 格式转换为可读的文本。
\item[pltotf]   将纯文本的 property list 转换为 TFM 格式。
\item[pooltype] 显示 \web{} 的 pool 文件。
\item[tangle]   将 \web{} 转换为 Pascal 代码。
\item[tex]   排版。
\item[tftopl]   将 TFM 格式转换为纯文本的 property list 格式。
\item[vftovp]   将虚拟字体格式转换为 virtual property list 格式。
\item[vptovf]   将 virtual property list 格式转换为虚拟字体格式。
\item[weave]   将 \web{} 转换为 \TeX。
\end{cmddescription}

\noindent 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 \Webc{} 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 \Webc{}。

所有的程序都接受这些 \GNU 标准的选项：
\begin{ttdescription}
\item[-{}-help] 显示基本使用说明。
\item[-{}-verbose] 显示详细的执行过程。
\item[-{}-version] 显示版本信息，然后退出。
\end{ttdescription}

所有的 \Webc{} 程序均使用 \KPS{} (\url{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 \TeX{} 文件的搜索。\Webc{} 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 \file{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。

\subsection{Kpathsea 路径搜索}
\label{sec:kpathsea}

我们首先介绍一下 \KPS{} 库的的通用路径搜索方式。

我们将目录名称称作\emph{路径元素}，而把用冒号或者分号分隔的路径元素列表称作\emph{搜索路径}。搜索路径可能是许多种来源的组合，比如在 \samp{.:/dir} 路径下搜索 \samp{my-file} 这个文件，\KPS{} 将逐个尝试路径中的每个元素：首先是 \file{./my-file}，然后是 \file{/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，\KPS{} 在非 Unix 系统下使用的路径分隔符可能不是冒号 (\samp{:}) 和斜杠 (\samp{/})。

在检查一个具体的路径元素 \var{p} 时，\KPS{} 首先检查是否有符合 \var{p} 的文件名数据库（见第~\pageref{sec:filename-database}~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 \var{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 \samp{!!} ，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 \KPS{} 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 \samp{.vf} 文件和 \TeX{} 用 \cs{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 \file{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 \samp{.vf} 文件后没有更新 \file{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，\KPS{} 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 \KPS{} 将搜索路径变换为一个或多个基本目录名的过程称为\emph{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 \samp{/} 或 \samp{./} 或 \samp{../} 起始，\KPS{} 将只检查该文件是否存在。

```
\ifSingleColumn
\else
\begin{figure*}
\verbatiminput{examples/ex5.tex}
\setlength{\abovecaptionskip}{0pt}
  \caption{一份示例性的配置文件的例子}
  \label{fig:config-sample}
\end{figure*}
\fi

\subsubsection{路径的来源}
\label{sec:path-sources}

搜索路径可能来自许多地方，\KPS{} 是按照下面的顺序查找的：

\begin{enumerate}
\item
  用户设置的环境变量，例如 \envname{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 \samp{latex}，那 \envname{TEXINPUTS.latex} 将比 \envname{TEXINPUTS} 优先级更高。
\item
  专门针对某个程序的配置文件，比如 \cmdname{dvips} 的 \file{config.ps} 里出现
```

\samp{S /a:/b} 这样一行。
\item   \KPS{} 配置文件 \file{texmf.cnf}，包含类似
  \samp{TEXINPUTS=/c:/d} 这样的一行（参见下面的解释）。
\item 编译时的缺省值。
\end{enumerate}
\noindent 你可以通过调试选项看到所有的这些值（参见第~\pageref{sec:debugging}~页的
``调试操作''）。

\subsubsection{配置文件}

\KPS{} 读入\emph{运行时配置文件} \file{texmf.cnf} 来获得搜索路径和其他定义。
而这个 \file{texmf.cnf} 存放的路径则是在 \envname{TEXMFCNF} 变量里定义的，
但我们不建议设置这个（或者任何）环境变量。

作为代替，标准的安装将产生文件 \file{.../2016/texmf.cnf}。如果你必须修改缺省
值（一般不需要），这是修改的地方。主配置文件是
\file{.../2016/texmf-dist/web2c/texmf.cnf}。你不应该修改后者，因为在发行版本被更
新后会丢失修改。

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

\begin{itemize}
\item
  以 \code{\%} 表示单行注释。
\item
  忽略空行。
\item
  行末的 \bs{} 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。
\item
  所有剩余的行格式如下：
\begin{alltt}
  \var{variable}[.\var{progname}] [=] \var{value}
\end{alltt}%
 \samp{=} 号和空白都是可选的。
\item
  \ttvar{variable}（变量）允许包含任何字符，除空白、\samp{=}、\samp{.}
  之外。不过只用 \samp{A-Za-z_} 是最保险的。
\item
  如果 \samp{.\var{progname}}（程序名）存在，则该定义只对正在运行的名叫
  \texttt{\var{progname}} 或 \texttt{\var{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。
\item \var{value}（值）允许任何 \code{\%} 与 \samp{@} 之外的字符出现。
  不支持在等号右侧使用 \code{\$\var{var}.\var{prog}} 这样的写法。
  如果在 Unix 下，\var{value} 中的 \samp{;}\ 字符会被转换为
  \samp{:}。如果你希望让 Unix，MS-DOS 和 Windows 里都用同一个
  \file{texmf.cnf}，这会很有用。
\item
  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。
\end{itemize}
展示上面所有内容的一段配置文件
\ifSingleColumn
如下：

\verbatiminput{examples/ex5.tex}
\else

见图~\ref{fig:config-sample}.
\fi

\subsubsection{路径展开}
\label{sec:path-expansion}

和 Unix shell 类似，\KPS{} 能够识别搜索路径中的特殊字符。比如一个复杂的路径
\verb+~$USER/{foo,bar}//baz+，将展开为这样的子目录：在 \texttt{\$USER}
的主目录下的 \file{foo} 或 \file{bar} 目录中，且包含 \file{baz}
文件或目录。这种展开将在下面解释。
%$
\subsubsection{默认展开}
\label{sec:default-expansion}

如果最高优先级（参见第~\pageref{sec:path-sources}~页的``路径来源''）
的搜索路径中包含一个\emph{额外的冒号}（即前置、后置或连续的冒号），\KPS{} 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

\begin{alltt}
> \Ucom{setenv TEXINPUTS /home/karl:}
\end{alltt}

而 \file{texmf.cnf} 里的 \code{TEXINPUTS} 值为

\begin{alltt}
  .:\$TEXMF//tex
\end{alltt}

则用于搜索的最终值为：

\begin{alltt}
  /home/karl:.:\$TEXMF//tex
\end{alltt}

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

\subsubsection{大括号展开}
\label{sec:brace-expansion}

大括号展开是一项有用的特性，其作用是把 \verb+v{a,b}w+ 这样的转换为
\verb+vaw:vbw+，允许嵌套使用。通过把 \code{\$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 \file{texmf.cnf} 里有下面的定义
（简化的例子）：
\begin{verbatim}
  TEXMF = {$TEXMFVAR,$TEXMFHOME,!!$TEXMFLOCAL,!!$TEXMFDIST}
\end{verbatim}
比如我们可以用它来设置 \TeX\ 的输入路径：
\begin{verbatim}
  TEXINPUTS = .;$TEXMF/tex//
\end{verbatim}
%$
的时候，检查完当前目录后，依次检查的路径是
\code{\$TEXMFVAR/tex}，\code{\$TEXMF/tex}，\code{\$TEXMFLOCAL/tex}
和 \code{\$TEXMF/tex}（后两个只在 \file{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 \code{\$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。

\subsubsection{子目录展开}
\label{sec:subdirectory-expansion}

在路径元素里的目录名称 \var{d} 后面接连使用两个或更多连续的斜杠，表示 \var{d} 的
所有子目录：首先是直接处于 \var{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\emph{不一定}的。

如果你在 \samp{//} 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 \samp{/a//b} 将展开为路径 \file{/a/1/b}，\file{/a/2/b}，
\file{/a/1/1/b} 等等，但不会展开为 \file{/a/b/c} 或 \file{/a/1}。

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

\subsubsection{特殊字符与其意义：简要说明}

下面的列表总结了 \KPS{} 配置文件中出现的特殊字符：

% need a wider space for the item labels here.
\newcommand{\CODE}[1]{\makebox[3em][l]{\code{#1}}}
\begin{ttdescription}
\item[\CODE{:}] 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。\par
\item[\CODE{;}] 非 Unix 系统下的路径分隔符（和 \code{:} 功能一样）。
\item[\CODE{\$}] 变量展开。
\item[\CODE{\string~}] 表示用户的个人主目录。
\item[\CODE{\char`\{...\char`\}}] Brace expansion.
\item[\CODE{//}] 子目录展开（可以出现在除路径开头外的任意位置）。
\item[\CODE{\%}] 注释的起始。
\item[\CODE{\bs}] 连接下一行的字符（以支持跨行的设置项）。
\item[\CODE{!!}] \emph{只}在数据库中搜索文件，\emph{不}搜寻磁盘。
\end{ttdescription}

\subsection{文件名数据库}
\label{sec:filename-database}

\KPS{} 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，\KPS{} 使用一个专门构建的纯文本``数据库''文件，这个文件叫做 \file{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 \file{aliases} 允许你给 \file{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。

\subsubsection{文件名数据库}
\label{sec:ls-R}

前边已经解释过，主文件名数据库的名称必须是 \file{ls-R}。你可以在每个需要搜索的
\TeX{} 目录树（缺省是 \code{\$TEXMF}）下放置一个这样的文件。\KPS{} 在
\code{TEXMFDBS} 指定的路径中寻找这些 \file{ls-R} 文件。

推荐使用发行版中包含的 \code{mktexlsr} 脚本来创建和维护 \samp{ls-R} 文件。各类
\samp{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：

\begin{alltt}

```
cd \var{/your/texmf/root} && \path|\|ls -1LAR ./ >ls-R
\end{alltt}
```

意味着你的系统的 \code{ls} 命令的输出格式必须正确 (\GNU \code{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 \code{cron} 来重建。

如果文件在数据库中找不到，缺省情况下 \KPS{} 会在磁盘上搜索。如果某个特定的路径元素
是以 \samp{!!} 起始的，就\emph{只会}在针对这一元素的数据库中查找而不搜索磁盘。

```
\subsubsection{kpsewhich: 独立的路径搜索}
\label{sec:invoking-kpsewhich}
```

\texttt{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 \code{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 \samp{mktex}\dots\
脚本中使用得非常多)。

```
\begin{alltt}
> \Ucom{kpsewhich \var{option}\dots{} \var{filename}\dots{}}
\end{alltt}
```
\ttvar{option} 处的选项可以用 \samp{-} 也可以用 \samp{-{}-} 来起始，并接受
任何不造成疑义的缩写。

\KPS{} 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 \samp{find} 程序来达到这个功能)。

下面介绍了一些比较常见的选项。

```
\begin{ttdescription}
\item[\texttt{-{}-dpi=\var{num}}]\mbox{}
  将解析度设置为 \ttvar{num}，这个选项只影响 \samp{gf} 文件和 \samp{pk}
  文件的查找。为了与 \cmdname{dvips} 兼容，提供 \samp{-D}
  这个同义的参数，默认值是 600。

\item[\texttt{-{}-format=\var{name}}]\mbox{}\\
  将查找的文件格式设置为 \ttvar{name}。默认情况下是通过文件名来猜测格式的。对于扩展名
有二义性的格式，比如 \MP{} 支持文件和 \cmdname{dvips} 配置文件，必须以 \KPS{} 已知
的名称指定格式，比如 \texttt{tex} 或 \texttt{enc files}。运行
  \texttt{kpsewhich -{}-help} 会显示格式的列表。

\item[\texttt{-{}-mode=\var{string}}]\mbox{}\\
  设置模式为 \ttvar{string}，只影响 \samp{gf} 和 \samp{pk}
  文件的查找。默认情况匹配所有的模式。
\item[\texttt{-{}-must-exist}]\mbox{}\\
  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
  \file{ls-R} 数据库里的内容。
\item[\texttt{-{}-path=\var{string}}]\mbox{}\\
  不通过文件名来猜测路径，沿 \ttvar{string} 指出的路径搜索
  (也使用冒号分隔)。支持 \samp{//} 和所有常见的展开方式。\samp{-{}-path} 选项
  和 \samp{-{}-format} 选项是互斥的。
\item[\texttt{-{}-progname=\var{name}}]\mbox{}\\
  将执行查找的程序名称设为 \texttt{\var{name}}。
  这会通过 \texttt{.\var{progname}} 特性影响搜索路径。
  缺省值是 \cmdname{kpsewhich}。
\item[\texttt{-{}-show-path=\var{name}}]\mbox{}\\
  显示用于查找 \texttt{\var{name}} 类型文件的路径。和 \samp{-{}-format}
  选项一样，可以使用扩展名 (\code{.pk}, \code{.vf}, 等等) 也可以使用全名。
```

```
\item[\texttt{-{}-debug=\var{num}}]\mbox{}\\
  将调试选项（等级）设置为 \texttt{\var{num}}。
\end{ttdescription}


\subsubsection{使用举例}
\label{sec:examples-of-use}

现在我们看看实际使用 \KPS{} 的例子。这里是一个简单的搜索：

\begin{alltt}
> \Ucom{kpsewhich article.cls}
   /usr/local/texmf-dist/tex/latex/base/article.cls
\end{alltt}
我们寻找的是 \file{article.cls} 文件。因为 \samp{.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 \optname{tex} 类型的文件（也就是要在（\TeX{} 源文件目录下查找）。
我们在 \samp{texmf-dist} 的 \file{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。
\begin{alltt}
> \Ucom{kpsewhich array.sty}
   /usr/local/texmf-dist/tex/latex/tools/array.sty
> \Ucom{kpsewhich latin1.def}
   /usr/local/texmf-dist/tex/latex/base/latin1.def
> \Ucom{kpsewhich size10.clo}
   /usr/local/texmf-dist/tex/latex/base/size10.clo
> \Ucom{kpsewhich small2e.tex}
   /usr/local/texmf-dist/tex/latex/base/small2e.tex
> \Ucom{kpsewhich tugboat.bib}
   /usr/local/texmf-dist/bibtex/bib/beebe/tugboat.bib
\end{alltt}

另外，最后一个是 \textsl{TUGBoat} 文章的 \BibTeX{} 参考文献数据库。

\begin{alltt}
> \Ucom{kpsewhich cmr10.pk}
\end{alltt}

\file{.pk} 类型的字体位图文件是给 \cmdname{dvips} 或者 \cmdname{xdvi}
这种显示程序提供的。因为 \TL{} 里没有预生成的 Computer Modern \samp{.pk}
字体，所以没有找到任何内容\Dash \TL{} 默认使用 Type~1 版本的。

\begin{alltt}
> \Ucom{kpsewhich wsuipa10.pk}
\ifSingleColumn   /usr/local/texmf-
var/fonts/pk/ljfour/public/wsuipa/wsuipa10.600pk
\else /usr/local/texmf-var/fonts/pk/ljfour/public/
...                          wsuipa/wsuipa10.600pk
\fi\end{alltt}

而这个字体（Washington 大学的一套注音字母表）就需要生成 \samp{.pk} 文件了。我们默认的
\MF{} 模式是 \texttt{ljfour}，基础解析度是 600\dpi{} (dots per inch)，所以会得到这样一个文
件。
\begin{alltt}
> \Ucom{kpsewhich -dpi=300 wsuipa10.pk}
\end{alltt}

一旦指明我们需要寻找的只是 300\dpi{} 的文件时（\texttt{-dpi=300}），就会发现系统里
没有符合要求的文件。这样 \cmdname{dvips} 或 \cmdname{xdvi} 会使用 \cmdname{mktexpk}
```

脚本去创建所需的 \texttt{.pk} 文件。

下面我们将注意力转向 \cmdname{dvips} 的头文件和配置文件。首先看看最常用的一个，\file{tex.pro} prologue 文件，然后检查通用配置文件 \file{config.ps} 和 \PS{} 字体映射文件 \file{psfonts.map}\Dash 从 2004 年开始，映射文件和编码文件都在 \dirname{texmf} 目录树下有其自己的搜索路径了。因为 \samp{.ps} 后缀可能会有二义，我们必须指明意思是 \optname{dvips config}。

```
\begin{alltt}
> \Ucom{kpsewhich tex.pro}
   /usr/local/texmf/dvips/base/tex.pro
> \Ucom{kpsewhich --format="dvips config" config.ps}
   /usr/local/texmf/dvips/config/config.ps
> \Ucom{kpsewhich psfonts.map}
   /usr/local/texmf/fonts/map/dvips/updmap/psfonts.map
\end{alltt}
```

这样我们可以更仔细地分析一下 URW Times 的 \PS{} 支持文件。依照标准的字体命名方案，这些文件的前缀是 \samp{utm}。首先查询的是配置文件，其中包含的是 map 文件的名称：

```
\begin{alltt}
> \Ucom{kpsewhich --format="dvips config" config.utm}
   /usr/local/texmf-dist/dvips/psnfss/config.utm
\end{alltt}
```

这个文件的内容是

```
\begin{alltt}
  p +utm.map
\end{alltt}
```

即指向 \file{utm.map} 文件，也就是我们下一步要找的。

```
\begin{alltt}
> \Ucom{kpsewhich utm.map}
   /usr/local/texmf-dist/fonts/map/dvips/times/utm.map
\end{alltt}
```

这个 map 文件定义了 URW 集合中的 Type~1 \PS{} 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）：

```
\begin{alltt}
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
\end{alltt}
```

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

```
\begin{alltt}
> \Ucom{kpsewhich utmr8a.pfb}
\ifSingleColumn   /usr/local/texmf-dist/fonts/type1/urw/times/utmr8a.pfb
\else   /usr/local/texmf-dist/fonts/type1/
```

```
... urw/utm/utmr8a.pfb
\fi\end{alltt}
```

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 \cmdname{kpsewhich} 只能告诉你找到的第一个文件。

```
\subsubsection{调试操作}
\label{sec:debugging}
```

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

```
\begin{ttdescription}
\item[\texttt{\ 1}] \texttt{stat} 调用（磁盘上的查询）。在 \file{ls-R}
  数据库及时更新的情况下几乎不会有什么输出。
\item[\texttt{\ 2}] 对散列表的引用（例如 \file{ls-R} 数据库，映射文件，配置文件）。
\item[\texttt{\ 4}] 文件的打开与关闭操作。
\item[\texttt{\ 8}] \KPS{} 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。
\item[\texttt{16}] 每个路径元素的目录列表（只在本地磁盘上搜索时有用）。
\item[\texttt{32}] 文件搜索。
\item[\texttt{64}] 变量的值。
\end{ttdescription}
```
等级 \texttt{-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 \cmdname{dvips} 程序中设置上述调试选项的组合，你就可以出 \cmdname{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 \KPS{} 库，你可以设置 \envname{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 \texttt{SET KPATHSEA_DEBUG_OUTPUT=err.log})。

让我们以一个简单的 \LaTeX{} 源文件 \file{hello-world.tex} 为例，其内容如下：
```
\begin{verbatim}
  \documentclass{article}
  \begin{document}
  Hello World!
  \end{document}
\end{verbatim}
```
这个小文件只使用了 \file{cmr10} 字体，我们可以看看 \cmdname{dvips} 是如何生成 \PS{} 文件的（我们希望使用 Type~1 版本的 Computer Modern 字体，所以使用了 \texttt{-Pcms} 参数）。
```
\begin{alltt}
> \Ucom{dvips -d4100 hello-world -Pcms -o}
\end{alltt}
```
此时我们将 \cmdname{dvips} 的调试等级 4（表示字体路径）和 \KPS\ 的路径元素展开组合到一起（参见 \cmdname{dvips} 参考手册。（稍作整理的）输出见图~\ref{fig:dvipsdbga}。
```
\begin{figure*}[tp]
\centering
\input{examples/ex6a.tex}
\caption{寻找配置文件}\label{fig:dvipsdbga}

\bigskip

\input{examples/ex6b.tex}
\caption{寻找 prolog 文件}\label{fig:dvipsdbgb}
```

```
\bigskip

\input{examples/ex6c.tex}
\caption{寻找字体文件}\label{fig:dvipsdbgc}
\end{figure*}

\cmdname{dvips} 启动后就开始搜寻其需要使用的文件。首先找到的是 \file{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 \file{ls-R} 文件名数据库（用于
优化文件搜索速度）和 \file{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 \cmdname{dvips} 去寻找它的通用
配置文件 \file{config.ps}，接下来查找个人定制文件 \file{.dvipsrc}（不过上述例子中并
未找到，显示 \emph{not found}）。最后 \cmdname{dvips} 找到了 Computer Modern
\PS{} 字体的配置文件 \file{config.cms}（因为执行 \cmdname{dvips} 时使用了
\texttt{-Pcms} 选项）。这个文件里包含了字体的 \TeX{} 名称，\PS{} 名称和文件名的对
应关系。

\begin{alltt}
> \Ucom{more /usr/local/texmf/dvips/cms/config.cms}
   p +ams.map
   p +cms.map
   p +cmbkm.map
   p +amsbkm.map
\end{alltt}
于是 \cmdname{dvips} 接下来寻找这些文件，再加上固定载入的，通用映射文件
\file{psfonts.map}（这个文件里包含常用 \PS{} 字体的映射，参见第
\ref{sec:examples-of-use}~节的最后一部分，讨论了 \PS{} 映射文件的处理）。

这时候 \cmdname{dvips} 向用户表示它的存在：
\begin{alltt}
This is dvips(k) 5.92b Copyright 2002 Radical Eye Software (www.radicaleye.com)
\end{alltt}
\ifSingleColumn
然后开始寻找 prolog 文件 \file{texc.pro}:
\begin{alltt}\small
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
\end{alltt}
\else
然后开始寻找 prolog 文件 \file{texc.pro}（见图~\ref{fig:dvipsdbgb}）。
\fi

找到所需文件后，\cmdname{dvips} 输出日期和时间，并告知我们它将生成
\file{hello-world.ps} 文件，需要用到 \file{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：
\begin{alltt}\small
TeX output 1998.02.26:1204' -> hello-world.ps
Defining font () cmr10 at 10.0pt
Font cmr10 <CMR10> is resident.
\end{alltt}
寻找并找到了 \file{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 \file{cmr10.pfb} 这个 Type~1 字体，并将它包含在输出文件中（见最后
一行）。
\begin{alltt}\small
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]
\end{alltt}

\subsection{运行时选项}

\Webc{} 另一项有用的特性是可以通过 \KPS{} 读入的运行时文件 \file{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:

\begin{ttdescription}
\item[\texttt{main\_memory}]
  总的可用内存数，以 word 为单位，\TeX{}, \MF{} 和 \MP\ 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 \texttt{hugetex.fmt}。
\item[\texttt{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。
\item[\texttt{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。
\item[\texttt{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
  \texttt{hash\_extra} 是 \texttt{50000}。
\end{ttdescription}

\noindent 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。

\section{致谢}

\TL{} 是在几乎所有 \TeX{} 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:

\begin{itemize*}

\item 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)，他们为所在
地区的 \TeX{} 社群提供了必备的技术和管理基础设施。请加入本地的 \TeX 用户组织！
(参见 \url{http://tug.org/usergroups.html}。)

\item CTAN 团队，值得一提的是 Robin Fairbairns 和 Rainer Sch\"opf。他们负
责分发 \TL{} 光盘镜像，为软件包更新提供支撑，\TL{} 正是基于这些软件包构建的。

\item Nelson Beebe，他为 \TL\ 提供了许多平台，自己也进行了详尽的测试，并在参考
文献上有无与伦比的贡献。

\item John Bowman，他对先进的图形程序 Asymptote 做了许多修改，使之能在 \TL{}
中工作。
```

\item Peter Breitenlohner 和 \eTeX\ 团队，他们创造了未来 \TeX\ 的稳定基础。
同时特别感谢 Peter，他为整个 \TL 中 \GNU\ autotools 的使用提供了主要的帮助。
Peter 在 2015 年 10 月去世，我们将持续工作以纪念他。

\item 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。

\item Thomas Esser，如果没有他优秀的 \teTeX{} 套件，\TL{} 根本不可能存在。

\item Michel Goossens，他一起编写了原始的文档。

\item 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，年复一年。Eitan 于 2009 年 6 月过早地离开了我们，我们希望把这份文档
献给它，以志纪念。

\item Hans Hagen，他对 \ConTeXt\ 格式 (\url{http://pragma-ade.com}) 做了
许多测试和修改，使之能够包含在 \TL{} 的框架下。

\item \Thanh, Martin Schr\"oder，和 pdf\TeX\ 团队，他们持续不断地改进 \TeX\
的功能。

\item Hartmut Henkel，他对 pdf\TeX\, Lua\TeX 等程序的开发起到了重要的贡献。

\item Taco Hoekwater，他对 \MP{} 和 (Lua)\TeX\ (\url{http://luatex.org})
重要的开发使之焕发新的活力。他还参与了将 \ConTeXt\ 融入 \TL{}、给
Kpathsea 添加多线程功能，等等许多工作。

\item Khaled Hosny，他对 \XeTeX, DVIPDFMx 和阿拉伯及其他字体的开发都有
巨大的贡献。

\item Pawe{\l} Jackowski，他创建了 Windows 下的安装程序 \cmdname{tlpm}，
和 Tomasz {\L}uczak，因为他的 \cmdname{tlpmgui} 被用在了以前的版本。

\item Akira Kakuto，以他的 W32TEX 发行版
(\url{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。

\item Jonathan Kew，他开发了非凡的 Xe\TeX{} 引擎并花了大量时间和精力集成到
\TL{} 中。以及 Mac\TeX 安装程序的早期版本，还有我们推荐的前端 \TeX{}works。

\item Dick Koch 他维护了 Mac\TeX\ (\url{http://tug.org/mactex}) 这个和
\TL{} 联系非常紧密的发行版本。

\item Reinhard Kotucha，他对 \TL{} 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，\texttt{getnonfreefonts} 脚本，等等。

\item Siep Kroonenberg，也因为他对 \TL{} 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。

\item Mojca Miklavec，为她在 \ConTeXt\ 上的工作。

\item Heiko Oberdiek，因为他的 \pkgname{epstopdf} 包和许多其他的工作，压缩
巨大的 \pkgname{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
\pkgname{hyperref} 宏包上优秀的工作。

\item Petr Ol\v{s}ak，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。

\item Toshio Oshima，他提供了 Windows 下的 \cmdname{dviout} 预览工具。

\item Manuel P\'egouri\'e-Gonnard，他对宏包更新、文档改进以及
\cmdname{texdoc} 开发上的努力。

\item Fabrice Popineau，他创建了 \TL{} 最早的 Windows 支持。

\item Norbert Preining，他是 \TL{} 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 \TL{} 工作（和 Frank K\"uster 一起），并提供了许
多建议。

\item Sebastian Rahtz，他是 \TL{} 的创始者，并维护了它许多年。Sebastian 在 2016
年 3 月去世，我们用持续的工作来纪念他。

\item Luigi Scarso，他坚持开发 MetaPost，Lua\TeX 等等。

\item Tomasz Trzeciak，他为 Windows 开发提供了广泛的帮助。

\item Vladimir Volovich，他很大地帮助解决了许多移植和维护的问题，尤其是令我们能够
将 \cmdname{xindy} 包含进来。

\item Staszek Wawrykiewicz，\TL{} 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。

\item Olaf Weber，他在以前几年对 \Webc{} 耐心的维护。

\item Gerben Wierda，他创建和维护了原来的 \MacOSX\ 支持。

\item Graham Williams，他是 \TeX\ Catalogue 的发起者。

\end{itemize*}

二进制版本的编译者：
Ettore Aldrovandi (\pkgname{i386-solaris}, \pkgname{x86_64-solaris}),
Marc Baudoin (\pkgname{amd64-netbsd}, \pkgname{i386-netbsd}),
Karl Berry (\pkgname{i386-linux}),
Ken Brown (\pkgname{i386-cygwin}, \pkgname{x86_64-cygwin}),
Simon Dales (\pkgname{armhf-linux}),
Akira Kakuto (\pkgname{win32}),
Dick Koch (\pkgname{universal-darwin}, \pkgname{x86_64-darwin}),
Nikola Le\v{c}i\'c (\pkgname{amd64-freebsd}, \pkgname{i386-freebsd}),
Mojca Miklavec (\pkgname{mipsel-linux}, \pkgname{sparc-solaris}),
Norbert Preining (\pkgname{alpha-linux}, \pkgname{x86_64-linux}),
Thomas Schmitz (\pkgname{powerpc-linux}),
Boris Veytsman (\pkgname{armel-linux}).
关于 \TL{} 编译进程的信息，请查阅 \url{http://tug.org/texlive/build.html}。

这份手册的翻译者：
Denis Bitouzé (法语),
Carlos Enriquez Figueras (西班牙语),
Jiang Jiang, Jinsong Zhao, Yue Wang, \& Helin Gai (中文),
Nikola Le\v{c}i\'c (塞尔维亚语),
Marco Pallante (意大利语),
Petr Sojka \& Jan Busa (捷克\slash 斯洛伐克语),
Boris Veytsman (俄语),
Staszek Wawrykiewicz (波兰语),
Uwe Ziegenhagen (德语)。\TL{} 文档的主页是

\url{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 \TeX，也感谢他将
\TeX\ 赠与全世界。

\section{发行历史}
\label{sec:history}

\subsection{过去}

1993 年末荷兰 \TeX{} 用户组开始为 MS-DOS 用户开发 4All\TeX{} \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\TeX{} \CD{}，同时 TUG
技术委员会工作组也开始设计 \emph{\TeX{} 目录结构} (\url{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 \textsl{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\TeX{} \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 \TL{} 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 \Webc{} 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
\texttt{texconfig} 脚本的辅助下，基于标准的 \Webc{} 来制作第二版的 \CD{}。
第 3 版的 \CD{} 基于 Olaf Weber 完成的 \Webc{} 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，\TL{} 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 \Webc{} (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。\TL{} 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 \texttt{language.dat}。

2002 年出现的第 7 版里显著的更新是添加了 \MacOSX{} 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。

\subsubsection{2003}

2003 年，在更新和增添持续不断到来的情况下，我们发现 \TL{} 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~\ref{sec:tl-coll-dists}
节，\p.\pageref{sec:tl-coll-dists})。此外：

\begin{itemize*}
\item 在 \LaTeX{} 团队的要求下，我们将 \cmdname{latex} 和 \cmdname{pdflatex}
      命令改为使用 \eTeX{} 引擎 (参见 \p.\pageref{text:etex})。
\item 包含了新的 Latin Modern 字体 (并推荐使用)。
\item 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)。
\item Windows 下的安装有很大改变，首次提供了基于 XEmacs 的集成环境。
\item Windows 下重要的辅助性程序 (Perl, Ghost\-script, Image\-Magick, Ispell)

      现在放在 \TL{} 的安装目录。
\item \cmdname{dvips}, \cmdname{dvipdfm} 和 \cmdname{pdftex}
      使用的字体映射文件现在通过 \cmdname{updmap} 这套新程序生成，并安装到
      \dirname{texmf/fonts/map} 目录下。
\item \TeX{}, \MF{}, 和 \MP{} 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 \verb|\write|），包括输出到文件、日志和终端上。也就是说，\emph{不再}使用
      \verb|^^| 标记来转换。在 \TL{}~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
      \verb|^^| 形式的输出，请将 \verb|texmf/web2c/cp8bit.tcx|
      文件改名。（后续版本将提供更简洁的方式来控制。）
\item 对文档作了大量更新。
\item 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：\TL{} 2003。
\end{itemize*}

\subsubsection{2004}

2004 年有许多改变：

\begin{itemize}

\item 如果你在本地安装的字体时涉及了 \filename{.map} 或 \filename{.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 \envname{TEXFONTMAPS} 变量中的路径设置，只在（所有 \filename{texmf}
目录树下的）\dirname{fonts/map} 子目录下搜索 \filename{.map}
文件。与之类似，\filename{.enc} 文件现在只在 \dirname{fonts/enc}
目录下搜索，根据 \envname{ENCFONTS} 变量中的路径设置。如果遇到有问题的文
件，\cmdname{updmap} 会提出警告。

关于这种搜索方式的其他信息，请参见 \url{http://tug.org/texlive/mapenc.html}。

\item 因为有人可能更愿意使用 \MIKTEX\ 而非 \Webc{} 系统，\TK\
现在包含了一套基于 \MIKTEX\ 的可安装 \CD，参见第~\ref{sec:overview-tl}~节
(\p.\pageref{sec:overview-tl})。

\item 在原来旧版本 \TL\ 中单一的 \dirname{texmf} 目录树被分拆为三
个：\dirname{texmf}, \dirname{texmf-dist}, 和 \dirname{texmf-doc}。参见
第~\ref{sec:tld}~节（\p.\pageref{sec:tld}）及各目录下的 \filename{README}
文件。

\item 所有 \TeX\ 输入文件现在统一收集到了 \dirname{texmf*} 下的
\dirname{tex} 子目录中，不再分散在各个 \dirname{tex}, \dirname{etex},
\dirname{pdftex}, \dirname{pdfetex} 目录。见~
\CDref{texmf-dist/doc/english/tds/tds.html\#Extensions}
{\texttt{texmf-dist/doc/english/tds/tds.html\#Extensions}}。

\item 辅助性脚本（并非直接提供给用户调用的）现在放在 \dirname{texmf*} 目录树下
新的 \dirname{scripts} 子目录中，并可以通过 \verb|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
\CDref{texmf-dist/doc/english/tds/tds.html\#Scripts}
{\texttt{texmf-dist/doc/english/tds/tds.html\#Scripts}}。

\item 几乎所有格式，都用 \filename{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 \verb|^^| 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

\OMEGA\ 相关的格式。缺省的情况几乎与 \TL\,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 \CDref{texmf-dist/doc/web2c/web2c.html\#TCX-files}
{\texttt{texmf-dist/doc/web2c/web2c.html\#TCX-files}}。
(另外，如果遇到 Unicode 输入，\TeX\ 可能会在显示错误上下文时输出半个字符，因为它是基于
字节流来处理输入的。)

\item \textsf{pdfetex} 现在是除 (plain) \textsf{tex} 外所有格式的默认引擎 (当然
以 \textsf{latex} 这种方式运行时它还是生成 DVI)。这样一来，至少 \textsf{pdftex}
的微调排版 (microtypographic) 技术可以在 \LaTeX, \ConTeXt 等格式中使用，另外 \eTeX\
的特性也包含在其中 (\OnCD{texmf-dist/doc/etex/base/})。

这还说明\emph{比以前任何时候都更有必要}使用 (对 plain 和 \LaTeX\ 都适用的)
\pkgname{ifpdf} 宏包或其类似代码，因为只检查 \cs{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 \cs{pdfoutput} 也可能已经定义。

\item pdf\TeX\ (\url{http://pdftex.org}) 新增了许多特性:

  \begin{itemize*}

  \item 可以使用 \cs{pdfmapfile} 和 \cs{pdfmapline} 来在单独文档内指定字体映射文件。

  \item 可以更方便地使用排版微调 (Microtypographic) 和字体延展 (font expansion)
  技术了。\\
  \url{http://www.ntg.nl/pipermail/ntg-pdftex/2004-May/000504.html}

  \item 原来使用专有格式的配置文件 \filename{pdftex.cfg} 里的选项现在都必须改用
  \TeX{} 原语来设置，通常放在 \filename{pdftexconfig.tex} 里面，不再支持
  \filename{pdftex.cfg} 的配置方式。每次修改 \filename{pdftexconfig.tex}
  之后都必须重新生成 \filename{.fmt} 文件。

  \item 参见 pdf\TeX\ 手册以了解更多信息: \OnCD{texmf-dist/doc/pdftex/manual/pdftex-
a.pdf}。

  \end{itemize*}

\item \cmdname{tex} (以及 \cmdname{mf} 和 \cmdname{mpost}) 中的 \cs{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:
\begin{verbatim}
\input "filename with spaces"   % plain
\input{"filename with spaces"} % latex
\end{verbatim}
参阅 \Webc{} 文档以了解更多信息: \OnCD{texmf-dist/doc/web2c}。

\item enc\TeX\ 的支持现在已被包含在 \Webc{} 中，因而所有 \TeX\ 程序都可以通过
\optname{-enc} 参数启用这一支持 \Dash \emph{前提是构建好了格式文件}。enc\TeX\
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 \OnCD{texmf-dist/doc/generic/enctex/} 和
\url{http://www.olsak.net/enctex.html}。

\item 提供了 Aleph 这套新的 \TeX{} 引擎，它将 \eTeX\ 和 \OMEGA\ 合并到了一起。
关于 Aleph 的部分信息可以在 \OnCD{texmf-dist/doc/aleph/base}
和 \url{http://www.tex.ac.uk/cgi-bin/texfaq2html?label=aleph} 找到。Aleph
的 \LaTeX 格式文件称做 \textsf{lamed}。

\item 最新发布的 \LaTeX\ 包含了是新版的 LPPL 授权协议\Dash 这一协议已被
Debian 首肯。\LaTeX\ 其他的更新请见 \OnCD{texmf-dist/doc/latex/base} 下的

\filename{ltnews} 文件。

\item 包含了一个叫做 \cmdname{dvipng} 的新程序，用于将 DVI 转换为
PNG 图像文件。参见 \url{http://www.ctan.org/pkg/dvipng}。

\item 我们在作者 (Claudio Beccari) 的同意下，将 \pkgname{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
(\url{http://mirror.ctan.org/tex-archive/fonts/greek/cbfonts})。

\item 去掉了 \cmdname{oxdvi}，改为只使用 \cmdname{xdvi}。

\item 不再为 \cmdname{tex}, \cmdname{mf}, 和 \cmdname{mpost} 程序创建
\cmdname{ini} 和 \cmdname{vir} 开头的命令链接，比如
\cmdname{initex}。\cmdname{ini} 的功能早在几年前就通过 \optname{-ini}
命令行参数提供了。

\item 去掉了 \textsf{i386-openbsd} 平台的支持。因为在 BSD Ports
系统中已经包含了 \pkgname{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。

\item 至少在 \textsf{sparc-solaris} 平台下，你必须设置好
\envname{LD_LIBRARY_PATH} 环境变量才能执行 \pkgname{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，\textsf{mips-irix} 平台下需要用到
MIPSpro 7.4 运行时库。

\end{itemize}

\subsubsection{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。

\begin{itemize}

\item 引入了新的 \cmdname{texconfig-sys}, \cmdname{updmap-sys}, 和
\cmdname{fmtutil-sys} 安装脚本，用于修改系统目录树下的配置。而原有的
\cmdname{texconfig}, \cmdname{updmap}, 和 \cmdname{fmtutil}
则用于修改针对单个用户的文件 (放在 \dirname{$HOME/.texlive2005}
目录下的)。

\item 增加了对应的 \envname{TEXMFCONFIG} 和 \envname{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
\filename{fmtutil.cnf} 和 \filename{updmap.cfg}
放到合适位置。不过还有一种方法是在 \filename{texmf.cnf} 里边重新定义
\envname{TEXMFCONFIG} 或 \envname{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~\pageref{sec:texmftrees}~页的第~\ref{sec:texmftrees}~节。

\item 虽然我们在上一年就已经使用了 \cmdname{pdfetex} 作为输出程序，但在它输出 \dvi\
格式时会禁用 \verb|\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
\verb|\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 \pkgname{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。

\item 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 \filename{empty.tcx} 来获得原有的
      \verb|^^| 表示方式。例如:

\begin{verbatim}
latex --translate-file=empty.tcx yourfile.tex
\end{verbatim}

\item 新增了用于转换 DVI 为 PDF 的 \cmdname{dvipdfmx} 程序，这是
      \cmdname{dvipdfm} 的一个比较活跃更新的版本 (我们仍然提供
      \cmdname{dvipdfm}，但不建议你继续使用)。

\item 新增了叫 \cmdname{pdfopen} 和 \cmdname{pdfclose} 的两个程序，用于控制
      Adobe Acrobat\slash 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 \cmdname{xpdf}, \cmdname{gv}, 和
      \cmdname{gsview}，都不会遇到这个问题。)

\item 为了保持一致性，将 \envname{HOMETEXMF} 和 \envname{VARTEXMF}
      环境变量分别更名为 \envname{TEXMFHOME} 和
      \envname{TEXMFSYSVAR}。还有一个针对单独用户的 \envname{TEXMFVAR}
      环境变量可用。参见上面的第一点。

\end{itemize}

\subsubsection{2006--2007}

2006--2007 年，\TL{} 的一个重大变化是增加了 \XeTeX{}，以 \texttt{xetex} 和
\texttt{xelatex} 程序的形式提供。请参见 \url{http://scripts.sil.org/xetex}。

\MP{} 也有可观的更新，并计划在未来实现更多的改进
(\url{http://tug.org/metapost/articles})，pdf\TeX{} 同样如此
(\url{http://tug.org/applications/pdftex})。

\TeX\ \filename{.fmt} (缓存格式) 文件和用于 MetaPost 和 \MF\ 的类似文件
现在存储在 \dirname{texmf/web2c} 的子目录中而不直接放在 \dirname{texmf/web2c}
目录下 (不过考虑到现有的 \filename{.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 \filename{tex}
或 \filename{pdftex} 或 \filename{xetex}。这个变化不会对日常使用带来任何
影响。

(plain) \texttt{tex} 程序不再通过读取 \texttt{\%\&} 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 \texttt{\%\&} 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 (\url{http://mirror.ctan.org})。

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

末了，2006 年五月 Thomas Esser 宣布他将停止 te\TeX{}
(\url{http://tug.org/tetex}) 的更新。这样一来，大家对 \TL{} 的兴趣大增，尤其是在
\GNU/Linux 发行版中。(\TL{} 提供了一套新的 \texttt{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。

\subsubsection{2008}

在 2008 年，整个 \TL{} 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 \filename{tlpkg/texlive.tlpdb} 这个纯文本文件中。

除了许多其他的功能外，我们终于可以通过 Internet 在安装后更新 \TL{} 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\ (\url{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 \TL 内部包含的 Perl。

新的 \cmdname{tlmgr} 脚本（第~\ref{sec:tlmgr}~节）是在初始安装后管理 \TL{} 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 \cmdname{tlmgr} 到来，禁用了 \cmdname{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 \cmdname{xindy} 索引生成程序 (\url{http://xindy.sourceforge.net})。

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

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

去除了 \filename{Times-Roman}, \filename{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 \pkgname{platex} 格式文件，以避免与日文 \pkgname{platex} 的命名冲突，\pkgname{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 \url{http://tug.org/TUGboat/Articles/tb29-2/tb92knut.pdf}。

\subsubsection{2009}

在 2009 年，为了充分利用 Lua\TeX\ 的 OpenType 支持等特性，Lua\AllTeX\ 的默认输出格式是 PDF。新增叫做 \code{dviluatex} 和 \code{dvilualatex} 的这两个命令会以 DVI 输出方式运行 Lua\TeX{}。Lua\TeX\ 的主页在 \url{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 \TypeI\ 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
\url{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
\url{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，\url{http://tug.org/texworks}。
设计这个跨平台前端的灵感来自于 \MacOSX\ 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
\url{http://asymptote.sourceforge.net}。

单独的 \code{dvipdfm} 程序已被 \code{dvipdfmx} 所替代，如果以 \code{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。\code{dvipdfmx}
包含了 \acro{CJK} 支持，并包含了多年以来在 \code{dvipdfm} 基础上的许多
修正。

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

一些更不起眼的更新: 我们现在使用 \pkgname{xz} 这套稳定的压缩方式来替代
原有的 \pkgname{lzma} (\url{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 \TL{} 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 \TL{}，包括 \CD\ 标签这些附属材料，
都在 \url{ftp://tug.org/historic/systems/texlive} 提供。

\subsubsection{2010}
\label{sec:2010news} % keep with 2010

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

在载入了 \LaTeX\ \code{graphics.cfg} 配置文件，而且输出的是 PDF 时，
pdf\AllTeX\ 现在\emph{自动地}将请求的封装 PostScript (EPS) 文件转换为
PDF，通过 \pkgname{epstopdf} 宏包。默认选项是为了避免覆盖任何手工创建
的 PDF 文件，但你也可以将 |\newcommand{\DoNotLoadEpstopdf}{}| (或
|\def...|) 放在 \cs{documentclass} 声明前以避免载入 \code{epstopdf}。
如果使用了 \pkgname{pst-pdf} 宏包，也不会载入它。要了解更多的细节，参
见 \pkgname{epstopdf} 宏包的文档 (\url{http://ctan.org/pkg/epstopdf-pkg})。

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

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

|openout_any| 设置。

\XeTeX\ 现在支持与 pdf\TeX\ 一样的 margin kerning。(Font expansion 现在
还不支持。)

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

包含了这些新的程序：用于排版日文的 p\TeX\ 引擎和相关实用工具；支持了
Unicode \BibTeX\ 的 \BibTeX{}U 程序；\prog{chktex} 实用工具
(\url{http://baruch.ev-en.org/proj/chktex})
用来检查 \AllTeX\ 文档；\prog{dvisvgm}
(\url{http://dvisvgm.sourceforge.net}) 是 DVI 到 SVG 格式转换器。

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

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

最后，\TK\ \DVD\ 上的 \TL{} 发行不能再直接执行了(听起来很奇怪)。
单张 \DVD\ 已经没有足够的空间了。一个优点是从物理 \DVD\ 的安装将会
快很多。

\subsubsection{2011}

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

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

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

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

恢复了 \cmdname{initex} 和 \cmdname{inimf} 程序。(但不包含其他的 \cmdname{ini*}
变种。)

\subsubsection{2012}

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

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

\code{pdftex} 和 \code{dvips} 等生成的输出文件现在可以超过 2 GB 了。

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

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

\filename{../texmf-local} 下现在也会出现一个 \code{texmf.cnf} 文件，
比如 \filename{/usr/local/texlive/texmf-local/web2c/texmf.cnf}。

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

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

\subsubsection{2013}

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

为了简化安装合并了许多小的语言集合。

\MP：加入对 PNG 输出和（IEEE 双精度）浮点数的原生支持。

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

\XeTeX\（见其主页了解更多信息）：
\begin{itemize}
\item 使用 HarfBuzz 库替代 ICU 进行字体排版。（仍然使用 ICU 来支持输入编码、双向
排版，以及可选的 Unicode 断行。）
\item 使用 Graphite2 和 HarfBuzz 来替代 SilGraphite 进行 Graphite 排版。
\item 在 Mac 上，使用 Core Text 替代（Apple 不再建议使用的）ATSUI。
\item 在名称相同的情况下有限使用 TrueType/OpenType 字体而不是 Type1 字体。
\item 修正偶尔出现的 \XeTeX\ 和 \code{xdvipdfmx} 字体查找不匹配的问题。
\item 支持 OpenType math 间距调整。
\end{itemize}

\cmdname{xdvi}：现在使用 FreeType 替代 \code{t1lib} 进行字体渲染。

\pkgname{microtype.sty}：对 \XeTeX\ 的部分支持（protrusion）和对 Lua\TeX\ 的支持
（protrusion, font expansion, tracking），已经其他改进。

\cmdname{tlmgr}：新的 \code{pinning} 操作以方便配置多个仓库；参见
\verb|tlmgr --help| 的对应章节，或者在线的
\url{http://tug.org/texlive/doc/tlmgr.html#MULTIPLE-REPOSITORIES}。

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

\subsubsection{2014}

2014 年我们收到了 Knuth 的又一个 \TeX\ 修正，这影响了所有的引擎，但可能唯一可见

的变化是在启动时恢复显示的 \code{preloaded format} 字符串。根据 Knuth 的说法，这个字符串现在反应的时在启动时\emph{应该被}默认载入的格式，而不是在二进制程序中预载入的未 dump 格式，该格式可能被很多方法覆盖。

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

Lua\TeX: 对于字体载入和断字 (hyphenation) 有明显的变化和修正。最大的增加时一个新的引擎, \code{luajittex} (\url{http://foundry.supelec.fr/projects/luajittex}) 和它的相关变体 \code{texluajit} 和 \code{texluajitc}。它使用的是一个即时编译的 Lua 编译器 (在 \textsl{TUGboat} 文章 \url{http://tug.org/TUGboat/tb34-1/tb106scarso.pdf} 有详
细介绍)。\code{luajittex} 还在开发中，所以并没有在所有平台提供，也比 \code{luatex} 要不稳定许多。我们及其作者都不建议使用它，除非为了试验 Lua 代码的即时编译 (JIT) 这样特殊的目的。

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

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

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

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

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

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

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

\subsubsection{2015}

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

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

pdf\TeX: 支持 JPEG Exif 和 JFIF; 就算在 \cs{pdfinclusionerrorlevel}

是负值时也不输出警告；同步到 \prog{xpdf}~3.04 版本。

Lua\TeX: 加入用于扫描 token 的 \pkgname{newtokenlib}；对 \code{normal} 随机数生成器等的 bug 修正。

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

MetaPost: 新的 \code{binary} numbersystem；新的支持日语的 \prog{upmpost} 和 \prog{updvitomp} 程序，类似 \prog{up*tex}。

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

基础设施：重新实现了 \prog{fmtutil} 脚本来按每个独立的目录树读取 \filename{fmtutil.cnf} 文件，类似 \prog{updmap}。Web2C \prog{mktex*} 脚本 (包括 \prog{mktexlsr}, \prog{mktextfm}, \prog{mktexpk}) 现在优先选择自己所在目录的程序，而不总选用当前 \envname{PATH} 里的。

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

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

\htmlanchor{news}
\subsection{当前版本---2016}
\label{sec:tlcurrent}

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

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

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

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

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

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

其他实用工具：

\begin{itemize}
\item \code{gregorio} 是一个新的程序，作为 \code{gregoriotex} 宏包的一部分用于 Gregorian 圣歌音乐的排版；它缺省就被包括在了 \code{shell_escape_commands} 中。

\item \code{upmendex} 是一个创建索引的程序，基本上和 \code{makeindex} 兼容，支持了 Unicode 排序，以及其他的一些修改。

\item \code{afm2tfm} 现在只将基于音调的高度调整上调；新选项 \code{-a} 忽略所有调整。

\item \code{ps2pk} 可以处理扩展 PK/GF 字体了。
\end{itemize}

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

基础架构：支持系统级别的 \code{tlmgr} 配置文件；校验包的完整性；如果有 GPG 还会校验网络更新的签名。(如果没有 GPG 则跳过这一步。)

平台：\code{alpha-linux} 和 \code{mipsel-linux} 被移除了。

\subsection{未来}

\emph{\TL{} 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。这个工作是由志愿者在其空闲时间完成的，所以总有更多值得做的地方。请参见 \url{http://tug.org/texlive/contribute.html}。

请把更正、建议或者提供帮助的意愿发送到：
\begin{quote}
\email{tex-live@tug.org} \\
\url{http://tug.org/texlive}
\end{quote}

\medskip
\noindent \textsl{祝你使用 \textrm{\TeX} 愉快！}

\section{翻译说明}

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

\begin{itemize}

\item package，视上下文，有时翻译为软件包，有时翻译为宏包。
\item format file，即 \TeX{} 程序一般都会预载入的 \texttt{.fmt} 文件。本文档中翻译为格式文件。
\item scheme，本文档中译为（安装）方案。
\item collection，本文档中译为（软件）集合。
\item 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。
\item 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。
\item binary，二进制文件，其实就是说可执行程序文件和库文件了。

\end{itemize}

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

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

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

2010 到 2015 年的简体中文版本都由 Jiang Jiang 翻译。

\end{document}
��译为（安装）方案。
\item collection, 本文档中译为（软件）集合。
\item 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。
\item 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。
\item binary, 二进制文件，其实就是说可执行程序文件和库文件了。

\end{itemize}

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