如何打包你的 LaTeX 宏包

Scott Pakin <scott+dtx@pakin.org>

21 January 2024

张泓知　　翻译

2024 年 1 月 29 日

摘　要

本教程适用于那些希望学习如何创建 `.ins` 和 `.dtx` 文件，以便分发他们自己编写的类和样式文件的高级 LaTeX 2ε 用户。

目　录

1 介绍

要求 我们假设您已经了解如何在 LaTeX 中编程。也就是说，您应该知道如何使用 \newcommand、\newenvironment，最好还懂一点 TeX。您还应该熟悉《面向宏包与类文件编写者的 LaTeX 2_ε》，它可以在 CTAN (http://www.ctan.org) 上获取，并且大多数 LaTeX 2_ε 发行版中都包含一个名为 clsguide.dvi 的文件。最后，您应该知道如何安装由 .dtx 文件和 .ins 文件组成的软件包。

术语 类文件 (.cls) 指定文档的基本格式：页面上的文本块大小和位置，结构元素如 \section 的排版，页眉和页脚样式等。一个文档中仅使用一个类文件（通过 \documentclass）。

样式文件 (.sty) 主要是一组宏和环境的定义。样式文件可以覆盖类文件的格式决策，提供新的功能，或更改现有功能。一个文档可以加载任意数量的样式文件（通过 \usepackage）。

一个或多个类文件或样式文件（例如，一个主样式文件使用 `\input` 或 `\RequirePackage` 来加载多个辅助文件）及其文档被称为 *宏包*。在本文档的其余部分，我们使用符号 "⟨*package*⟩" 来表示你的宏包的名称。警告：在 LaTeX 社区中，有时也使用术语 "宏包" 来指 "样式文件"。您可能需要从上下文中判断使用的是哪个定义。

动机 一个包的重要部分包括代码、代码的文档和用户文档。使用 Doc 和 DocStrip 程序，可以将这三者合并为一个单一的，*带文档的 LaTeX* （`.dtx`）*文件*。`.dtx` 文件的主要优势在于，它允许您使用任意的 LaTeX 构造来注释您的代码。因此，宏、环境、代码段、变量等都可以使用表格、图形、数学公式和字体变化来解释。代码可以使用 LaTeX的分段命令进行组织。Doc甚至可以生成一个统一的索引，对宏定义（在 LaTeX 代码中）和宏描述（在用户文档中）进行索引。这种注重为代码编写详细的、漂亮排版的注释方法——本质上将程序视为描述一组算法的书——被称为 *文学编程* [2]，并自早期的 TeX 开始就被使用。

本教程将教您如何编写基本的 `.dtx` 文件和操作它们的 `.ins` 文件。虽然与《LaTeX Companion》的第 14 章存在许多重叠 [1]，但本文档结构更像是一步一步的教程，而《LaTeX Companion》更像是参考资料。此外，本教程展示了如何编写一个单一文件，既作为文档又作为驱动文件，这是 Doc 系统的一种更典型的用法，而不是使用分开的文件。

2 `.ins` 文件

为了准备一个包用于发布，第一步是编写一个*安装*（`.ins`）*文件*。安装文件从 `.dtx` 文件中提取代码，使用 DocStrip 去掉注释和文档，然后输出一个 `.sty` 文件。好消息是，`.ins` 文件通常相当简短，并且在一个包到另一个包之间没有明显变化。

`.ins` 文件通常以注释开始，指定版权 和许可信息：

```
%%
%% Copyright (C) ⟨year⟩ ⟨your name⟩
```

```
%%
%% This file may be distributed and/or modified under the
%% conditions of the LaTeX Project Public License, either
%% version 1.3 of this license or (at your option) any later
%% version.  The latest version of this license is in:
%%
%%    http://www.latex-project.org/lppl.txt
%%
%% and version 1.3c or later is part of all distributions of
%% LaTeX version 2008-05-04 or later.
%%
```

LaTeX 项目公共许可证（LPPL）是大多数包——以及 LaTeX 本身——所使用的许可证。当然，您可以根据您想要的任何许可证发布您的包；LPPL 只是 LaTeX 包中最常见的许可证。LPPL 规定用户可以对您的包做任何事情——包括出售它，并且无需向您支付任何费用。唯一的限制是他必须为您的工作给予您信用，并且他必须清晰地将修改过的代码明确标识为修改版本，以避免版本混淆。

下一步是加载 DocStrip:

```
\input docstrip.tex
```

`\keepsilent`

默认情况下，DocStrip 会详细列出其活动情况。这些消息并不是特别有用，所以大多数人会将其关闭：

```
\keepsilent
```

`\usedir {⟨directory⟩}`

系统管理员可以指定所有与 TEX 相关文件应安装在其下的基本目录，例如 /usr/share/texmf。（请参阅 DocStrip 手册中的“\BaseDirectory”。）.ins 文件指定其文件相对于该目录应安装的位置。以下是典型的设置：

```
\usedir{tex/latex/⟨package⟩}
```

```
\preamble
⟨text⟩
\endpreamble
```

接下来的步骤是指定一个 *preamble*，即将写入到每个生成文件顶部的一段注释：

```
\preamble

This is a generated file.

Copyright (C) ⟨year⟩ ⟨your name⟩

This file may be distributed and/or modified under the
conditions of the LaTeX Project Public License, either
version 1.3 of this license or (at your option) any later
version.  The latest version of this license is in:

   http://www.latex-project.org/lppl.txt

and version 1.3c or later is part of all distributions of
LaTeX version 2008-05-04 or later.

\endpreamble
```

前述的前言会导致 ⟨*package*⟩.sty 文件开头如下：

```
%%
%% This is file `⟨package⟩.sty',
```

```
%% generated with the docstrip utility.
%%
%% The original source files were:
%%
%% ⟨package⟩.dtx  (with options: `package')
%%
%% This is a generated file.
%%
%% Copyright (C) ⟨year⟩ ⟨your name⟩
%%
%% This file may be distributed and/or modified under the
%% conditions of the LaTeX Project Public License, either
%% version 1.3 of this license or (at your option) any later
%% version.  The latest version of this license is in:
%%
%%    http://www.latex-project.org/lppl.txt
%%
%% and version 1.3c or later is part of all distributions of
%% LaTeX version 2008-05-04 or later.
%%
```

`\generate {\file {⟨style-file⟩} {\from {⟨dtx-file⟩} {⟨tag⟩}}}`

现在我们来到一个 .ins文件中最重要的部分：指定从 .dtx文件生成哪些文件。以下告诉 DocStrip 从 ⟨*package*⟩.dtx 中仅提取标记为“package”的部分，生成 ⟨*package*⟩.sty。（如何标记 .dtx 文件的部分在第 3节中描述。）

```
\generate{\file{⟨package⟩.sty}{\from{⟨package⟩.dtx}{package}}}
```

\generate 可以从给定的 .dtx 文件中提取任意数量的文件。它甚至可以从多个 .dtx 文件中提取单个文件。详细信息请参阅 DocStrip 手册。

`\Msg {⟨text⟩}`

`.ins` 文件的下一部分包括命令，用于向用户输出消息，告诉他需要安装哪些文件，并提醒他如何生成用户文档。以下一组 `\Msg` 命令是典型的：

```
\obeyspaces
\Msg{*****************************************************}
\Msg{*                                                   *}
\Msg{* To finish the installation you have to move the  *}
\Msg{* following file into a directory searched by TeX: *}
\Msg{*                                                   *}
\Msg{*     ⟨package⟩.sty                                 *}
\Msg{*                                                   *}
\Msg{* To produce the documentation run the file        *}
\Msg{* ⟨package⟩.dtx through LaTeX.                      *}
\Msg{*                                                   *}
\Msg{* Happy TeXing!                                    *}
\Msg{*                                                   *}
\Msg{*****************************************************}
```

请注意使用 `\obeyspaces` 来阻止 TEX 合并多个空格为一个。

`\endbatchfile`

最后，我们告诉 DocStrip 已经到达 `.ins` 文件的末尾：

```
\endbatchfile
```

附录 A.1 列出了一个完整的 `.ins` 框架文件。附录 A.2 类似，但包含了一些微小的修改，旨在生成一个类（`.cls`）文件，而不是样式（`.sty`）文件。

3 `.dtx` 文件

一个 `.dtx` 文件包含了包的有注释源代码和用户文档。通过运行 `latex` 命令来处理一个 `.dtx` 文件，可以排版出用户文档，通常还包括一个漂亮排版的有注释 源代码版本。

由于一些 Doc 的技巧，一个 .dtx 文件实际上被评估了两次。第一次，只评估了一小部分 LaTeX 驱动代码。第二次，*comments* 在 .dtx 文件中被评估，就好像它们前面没有 "%"。这可能会导致写 .dtx 文件时产生许多混乱，并偶尔导致一些笨拙的构造。幸运的是，一旦 .dtx 文件的基本结构就位，填写代码就相当简单。

3.1 序言

.dtx 文件通常以版权和许可的注释开始：

```
% \iffalse meta-comment
%
% Copyright (C) ⟨year⟩ ⟨your name⟩
%
% This file may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either
% version 1.3 of this license or (at your option) any later
% version.  The latest version of this license is in:
%
%    http://www.latex-project.org/lppl.txt
%
% and version 1.3c or later is part of all distributions of
% LaTeX version 2008-05-04 or later.
%
% \fi
```

由于第二次处理 .dtx 文件时，行首的 % 字符会被忽略，所以需要使用 \iffalse 和 \fi。为了防止版权/ 许可被解释为 LaTeX 代码，我们必须将其用 \iffalse…\fi 括起来。在"\iffalse"后添加"meta-comment"只是一种约定，表示这个注释是为人类阅读而非 Doc、DocStrip 或 LaTeX 的。

```
\NeedsTeXFormat {⟨format-name⟩} [⟨release-date⟩]
\ProvidesPackage {⟨package-name⟩} [⟨release-info⟩]
```

接下来的几行同样被 `\iffalse…\fi`包围，以防止在第二次通过 `.dtx` 文件时被 `latex` 处理。不过，这些行不是为人类读者准备的，而是为了 DocStrip （因此没有“`meta-comment`”）：

```
% \iffalse
%<package>\NeedsTeXFormat{LaTeX2e}[2023-11-01]
%<package>\ProvidesPackage{⟨package⟩}
%<package>    [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨description⟩]
%
```

（我们很快就会遇到 `\fi`。）

还记得 `.ins` 文件（第 6 页）中的 `\generate` 行吗？它以标签“`package`”结束。这告诉 DocStrip 将以“`%<package>`”开头的行写入到 `.sty` 文件中，并在此过程中剥离“`%<package>`”。因此，我们的样式文件 `.sty` 将在头部注释之后包含以下代码：

```
\NeedsTeXFormat{LaTeX2e}[2023-11-01]
\ProvidesPackage{⟨package⟩}
   [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨description⟩]
```

比如：

```
\NeedsTeXFormat{LaTeX2e}[2023-11-01]
\ProvidesPackage{skeleton}
   [2002-03-25 v1.0 .dtx skeleton file]
```

`\NeedsTeXFormat` 行确保宏包不会在除 LaTeX 2_ε 外的其他 TeX 格式下运行。可选的日期参数导致 `latex` 在LaTeX 2_ε的版本（可以通过LaTeX 2_ε的 `\fmtversion` 宏找到）早于宏包测试版本时输出警告消息：

```
LaTeX Warning: You have requested release `2239-09-30' of LaTeX,
               but only release `2023-11-01' is available.
```

在 \ProvidesPackage 行中的日期和版本字符串被 Doc用于设置 \filedate 和 \fileversion 宏。请注意日期格式；在LaTeX 2ε中，*YYYY-MM-DD* 被广泛使用，因此在您的宏包中也应该使用。[1]

```
\EnableCrossrefs
\CodelineIndex
\RecordChanges
\DocInput {⟨filename⟩}
```

接下来是 .dtx 文件中唯一不被注释掉的部分（即每行不以 "%" 开头）：

```
%<*driver>
\documentclass{ltxdoc}
\usepackage{⟨package⟩}
\EnableCrossrefs
\CodelineIndex
\RecordChanges
\begin{document}
  \DocInput{⟨package⟩.dtx}
\end{document}
%</driver>
% \fi
```

前述的代码块是 latex 在第一次处理 .dtx 文件时所评估的内容。现在我们逐行来看这段代码：

1. 将代码放置在 "%<*driver>" 和 "%</driver>" 之间是 DocStrip 的一种简写，表示在每一行前加上 "%<driver>"。这标示了 Doc 的驱动代码。

2. \documentclass 几乎总是应该使用 ltxdoc，因为这会加载 Doc 并提供一些有用的宏来格式化程序文档。

3. 你应该总是使用 \usepackage 导入你的样式文件。如果不这样做，Doc 将看不到宏包的 \ProvidesPackage 行，也就无法设置 \filedate 和

[1] LaTeX 2ε的旧版本使用 *YYYY/MM/DD* 格式，但最新的宏包/类作者指南 [3] 规定从现在开始应该使用 *YYYY-MM-DD*。

\fileversion （参见第 12 页）。这也是你应该使用 \usepackage 导入任何用于排版用户文档的其他宏包的地方（与你的宏包需要的宏包不同）。

4. \EnableCrossrefs 告诉 Doc 您希望它为您的代码构建索引——通常是个好主意。另一种选择是 \DisableCrossrefs，它可以在处理速度上稍微提升一些，但影响微乎其微。

5. \CodelineIndex 告诉 Doc 索引应该引用程序行号而不是页码（另一种选择是 \PageIndex）。\CodelineIndex 使得索引条目更易于查找，但以索引的自洽性稍有损失（因为宏和环境的描述总是按页码索引）。索引，不过，会以一条说明性的注释开始。

6. 在第 12 页，讨论了如何记录包每个版本的更改。\RecordChanges 告诉 Doc 应该保留并汇总日志条目。

7. 在 \begin{document} 和 \end{document} 之间应该只有一个命令：一个 \DocInput 调用，用于 .dtx 文件自身的输入。这使得主文件可以通过 \DocInput 来输入多个文件，从而生成一个单一文档，涵盖了多个包但包含了一个统一的索引。主文档文件在第 on page 24 页有描述。

\OnlyDescription

在前言（即 \begin{document} 之前）有时会出现的另一个命令是 \OnlyDescription，它告诉 Doc 仅排版用户文档，而不是包的代码或注释。最好通常省略 \OnlyDescription（或将其注释掉）。用户始终可以手动添加它，甚至可以通过将以下内容添加到他的 ltxdoc.cfg 文件中，为所有 .dtx 文件启用 \OnlyDescription：

```
\AtBeginDocument{\OnlyDescription}
```

本节剩余部分涵盖了 latex 对 .dtx 文件的第二次处理。因此，所有随后的示例都以百分号开头。

\changes {⟨*version*⟩} {⟨*date*⟩} {⟨*description*⟩}

在第 11 页我们了解到 Doc 有一个记录包变更的机制。命令是“\changes{⟨*version*⟩}{⟨*date*⟩}{⟨*description*⟩}”，通常用 \changes 来记录包的初始版本和创建日期:

```
% \changes{v1.0}{2002/03/25}{Initial version}
```

\changes 命令的一个好处是它知道自己是在宏/环境定义的内部使用还是顶层使用。如图 1 所示，顶层变更以“General:”为前缀，而内部变更以包围它的宏或环境的名称为前缀。

Change History

v1.0

General: Top-level comment 1

v1.2j

myMacro: Internal macro comment 5

Figure 1: Sample change history

\GetFileInfo {⟨*style-file*⟩}

\filedate
\fileversion
\fileinfo

接下来，我们告诉 Doc 解析 \ProvidesPackage 命令（第 9 页），依次调用 \ProvidesPackage 参数的三个组成部分，分别是“\filedate”、“\fileversion”和“\fileinfo”:

```
% \GetFileInfo{⟨package⟩.sty}
```

例如，\ProvidesPackage 的示例（见第 on page 9 页）将被解析如下:

`\filedate`	≡	2002-03-25
`\fileversion`	≡	v1.0
`\fileinfo`	≡	.dtx skeleton file

`\DoNotIndex {`⟨*macro-name , ...*⟩`}`

在生成索引时，Doc 通常会索引代码中的每个控制序列（即反斜线开头的单词或符号）。这种自动化水平的问题在于，许多控制序列在理解代码的角度来看并不重要。例如，读者可能不想看到每个使用了 `\if`、`\the`、`\let`、`\begin` 或其他众多控制序列的地方。

正如其名称所示，`\DoNotIndex` 命令给出了一个 不应被索引的控制序列列表给 Doc。`\DoNotIndex` 可以使用任意次数，并且每次调用可以接受任意数量的控制序列名称：

```
% \DoNotIndex{\#,\$,\%,\&,\@,\\,\{,\},\^,\_,\~,\ }
% \DoNotIndex{\@ne}
% \DoNotIndex{\advance,\begingroup,\catcode,\closein}
% \DoNotIndex{\closeout,\day,\def,\edef,\else,\empty,\endgroup}
```

⋮

3.2 用户文档

我们终于可以开始编写用户文档了。典型的开头如下：

```
% \title{The \textsf{⟨package⟩} package\thanks{This document
%   corresponds to \textsf{⟨package⟩}~\fileversion,
%   dated~\filedate.}}
% \author{⟨your name⟩ \\ \texttt{⟨your e-mail address⟩}}
%
% \maketitle
```

当然标题可以更有创意，但请注意，使用 `\textsf` 排版包名是常见的做法，而使用 `\thanks` 来指定包的版本和日期。这样做带来了文学化编程的一

个优点：每当你改变包的版本（作为 \ProvidesPackage 的可选第二参数），用户文档会相应更新。当然，你仍然需要手动确保用户文档准确描述了更新后的包。

以你处理任何 LaTeX 文档的方式撰写用户文档，不过每行都要以 "%" 开头。请注意，ltxdoc 文档类派生自 article，因此顶级分段命令是 \section，而不是 \chapter。

\DescribeMacro {⟨*macro*⟩}
\DescribeEnv {⟨*environment*⟩}

Doc 提供了一些命令来帮助格式化用户文档。如果你在段落中包含 "\DescribeMacro{⟨*macro*⟩}"[2]，Doc 会将 "⟨*macro*⟩" 放在边距中，方便读者查看。同时，Doc 还会将 ⟨*macro*⟩ 加入索引，并格式化相应的页码，以表明这是描述该宏的位置（而不是宏在源代码中定义的地方）。

\DescribeEnv 是描述环境的类似命令。\DescribeMacro 和 \DescribeEnv 都可以在同一段落中多次使用。

\NewDocElement [⟨*options*⟩] {⟨*element-name*⟩} {⟨*env-name*⟩}

在 LaTeX文档中，最常见的是记录宏和环境，\NewDocElement 方便地用于记录任意宏包组件。它定义了类似于\DescribeMacro 和 \DescribeEnv 的 \Describe⟨*element-name*⟩ 宏。然后，你可以使用 \begin{⟨*env-name*⟩}…\end{⟨*env-name*⟩} 来标记新元素的实例，类似于描述在 on page 18上的 \begin{macro}…\end{macro} 和 \begin{environment}…\end{environment}。⟨*options*⟩ 参数指定了新元素是 macrolike（以反斜杠开头）还是 envlike（不以反斜杠开头），等等。

\marg {⟨*argument*⟩}
\oarg {⟨*argument*⟩}
\parg {⟨*argument*⟩}
\meta {⟨*text*⟩}

[2] "⟨*macro*⟩" 应包括反斜线。

ltxdoc 文档类提供了三个命令来帮助排版宏和环境的语法（见表 1）。\marg 用于排版必选参数，\oarg 用于排版可选参数，\parg 用于排版图形模式参数。这三个命令都使用 \meta 来排版参数内容。\meta 也可单独使用。例如，“This needs a \meta{dimen}.” 会排版为 “This needs a ⟨*dimen*⟩.”。

Table 1: Argument-formatting commands

Command	Result
\marg{text}	{⟨*text*⟩}
\oarg{text}	[⟨*text*⟩]
\parg{text}	(⟨*text*⟩)

除了这些命令外，Doc 通过自动加载 shortvrb 包，便利地简化了宏描述的排版。shortvrb 允许你使用 |…|，作为方便的简写来表示 \verb|…|。例如，“|\mymacro| \oarg{pos} \marg{width} \marg{text}” 的排版如下：

```
\mymacro [⟨pos⟩] {⟨width⟩} {⟨text⟩}
```

与 \verb 类似，|…| 简写在 \footnote 或其他易碎的宏中无法使用。

3.3 代码和评论

```
\MaybeStop {⟨text⟩}
\Finale
```

包的源代码位于\MaybeStop[3] 和\Finale之间。\MaybeStop带有一个参数，即在代码后要排版的文本块。如果指定了\OnlyDescription（第 11页），则\MaybeStop之后将不会输出任何内容，包括\Finale后面的文本。因此，\MaybeStop的⟨*text*⟩参数是提供一个文本块的机制，无论是否排版代码列表，都应该输出该文本块。通常包括参考文献部分和/或以下两个命令之一：

[3]在早期版本的 Doc中，\MaybeStop被称为\StopEventually。后者仍然被定义，但已被弃用。

\PrintChanges
\PrintIndex

\PrintChanges 生成一个名为“变更历史”的无编号章节（见图 1 on page 12）。变更历史部分汇总了 .dtx文件中所有 \changes命令的各版本修改，这样就能轻松跟踪每个版本的变更内容。

\PrintChanges 使用了 LaTeX的术语表机制。在 ⟨*package*⟩.dtx 上运行 latex 会在 ⟨*package*⟩.glo 中生成变更历史数据。要生成排版的变更历史（⟨*package*⟩.gls），用户应该按以下方式运行 makeindex程序：

```
makeindex -s gglo.ist -o ⟨package⟩.gls ⟨package⟩.glo
```

\PrintIndex 生成一个名为“索引”的无编号章节。索引会自动包含文档中使用、定义或描述的所有宏和环境条目。所有环境还会额外列在“环境”下。表 2 演示了各种条目的格式。在该表中，“27”指的是页码，“123”指的是行号。[4] 请注意，只有在文档包含代码列表（即未指定 \OnlyDescription）时，宏/环境的定义和使用才会包含在索引中。

Table 2: Formatting of entries in the index

Item	Function	Formatting in index
Macro	Used	\myMacro 123
Macro	Defined	\myMacro 123
Macro	Described	\myMacro *27*
Environment	Defined	myEnv (environment) 123
Environment	Described	myEnv (environment) *27*
Explicit \index	—	myItem 27

对于显式 \index命令的默认格式化使用了罗马页码。这会导致混淆，因为在包源代码中罗马页码通常表示行号。解决方案是对\index命令指定“usage”格式，以使页码以斜体排版：

```
\index{explicit indexing|usage}
```

[4]如果未使用 \CodelineIndex（第 10 页），则“123”将指的是页码。

在 ⟨*package*⟩.dtx 上运行 latex 会在 ⟨*package*⟩.idx 中生成索引数据。要生成排版的索引(⟨*package*⟩.ind),用户应该按以下方式运行 makeindex程序:

```
makeindex -s gind.ist -o ⟨package⟩.ind ⟨package⟩.idx
```

代码索引是文学化编程的一个很好的“附加值”。它几乎不需要额外的工作量，并极大地帮助代码维护人员找到宏定义以及了解包依赖的其他宏。

```
\begin{macrocode}
⟨code⟩
\end{macrocode}
```

\begin{macrocode} 和 \end{macrocode} 之间列出的代码片段会原样提取到 .sty 文件中。排版时，代码片段会显示带有行号计数器，以便于指定特定行。以下是关于 macrocode 环境的一些要点:

1. “%” 和 “\begin{macrocode}” 或 “\end{macrocode}” 之间必须有精确地四个空格。否则，Doc 将无法检测到代码片段的结束。[5]
2. 在 \begin{macrocode}…\end{macrocode} 中的代码行不应以 “%” 开头。代码将按原样写入到 .ins 文件中，不会删除 %。

以下是一个示例代码片段。它恰好是一个完整的宏定义，但这并非必须；任何 LaTeX 代码片段都可以出现在 macrocode 环境中。

```
%    \begin{macrocode}
\newcommand{\mymacro}{This is
  a \LaTeX{} macro.}
%    \end{macrocode}
```

Doc 将前面的代码片段格式化为以下内容:

[5]趣闻：只有 \end{macrocode} 需要这种精确的间距，而且仅用于文档的排版。尽管如此，在 \begin{macrocode} 中使用 “%␣␣␣␣” 是个好习惯。

```
\newcommand{\mymacro}{This is
  a \LaTeX{} macro.}
```

请注意，行号在整个程序中是唯一的（而不是在每一页顶部重置）。如果在包含前述 \mymacro 定义的 .dtx 文件中使用 \PrintIndex，索引将自动包含 \newcommand、\mymacro 和 \LaTeX 的条目，除非其中任何一个被 \DoNotIndex排除。

```
\begin{macro}{⟨macro⟩}
    ⋮
\end{macro}

\begin{environment}{⟨environment⟩}
    ⋮
\end{environment}
```

macro 和 environment 环境用于界定完整的宏或环境定义。macro /environment 环境通常包含一个或多个 macrocode 环境，其中穿插着代码文档。以下是 macrocode示例的更完整版本，见 on the preceding page。

```
% \begin{macro}{\mymacro}
% We define a trivial macro, |\mymacro|, to illustrate
% the use of the |macro| environment.
%    \begin{macrocode}
\newcommand{\mymacro}{This is
  a \LaTeX{} macro.}
%    \end{macrocode}
% \end{macro}
```

以下是排版后的版本：

\mymacro　We define a trivial macro, \mymacro, to illustrate the use of the macro environment.

```
\newcommand{\mymacro}{This is
  a \LaTeX{} macro.}
```

Doc 在边距中排版宏/环境名称，以增加可见性。此外，Doc 也会将相应条目添加到索引中（参见表 2 on page 16，了解这些条目的格式示例）。请注意，`\begin{macro}…\end{macro}` 不是必需的来指示宏定义。它也可以用于指示其它控制序列的定义，例如计数器、长度和盒子：

```
% \begin{macro}{myCounter}
% This is an example of using the |macro| environment to format
% something other than a macro.
%    \begin{macrocode}
\newcounter{myCounter}
%    \end{macrocode}
% \end{macro}
```

`macro` 和 `environment` 环境可以嵌套。这种能力不仅对于定义其他宏的宏很有用，而且在定义一组共享描述的相关数据类型时也很有用：

```
% \begin{macro}{\thingheight}
% \begin{macro}{\thingwidth}
% \begin{macro}{\thingdepth}
% These lengths keep track of the dimensions of our |\thing|
% box.  (Actually, we're just trying to show how to nest
% |macro| environments.)
%    \begin{macrocode}
\newlength{\thingheight}
\newlength{\thingwidth}
\newlength{\thingdepth}
%    \end{macrocode}
% \end{macro}
% \end{macro}
% \end{macro}
```

为了方便起见，Doc的最新版本允许一个`\begin{macro}`同时列出多个要定义的宏：

```
% \begin{macro}{\thingheight,\thingwidth,\thingdepth}
% These lengths keep track of the dimensions of our |\thing|
% box.  (Actually, we're just trying to show how to nest
% |macro| environments.)
%    \begin{macrocode}
\newlength{\thingheight}
\newlength{\thingwidth}
\newlength{\thingdepth}
%    \end{macrocode}
% \end{macro}
```

通常应避免没有描述的 macro 环境，因为格式有些丑陋：宏名称单独出现在一行上，位于一个“空白”描述的左侧，但代码直到下一行才开始。

在\begin{macro}…\end{macro}或\begin{environment}…\end{environment}块内可以有多个 macrocode环境。这是代码可以被内部注释的机制，用于宏/环境。（在 macrocode块内使用“%”注释被认为是不良风格。）以下是一个非平凡的宏的注释示例：

```
% \begin{macro}{\complexMacro}
% Pretend that this is a very complex macro that needs
% to have its various pieces documented.
%    \begin{macrocode}
\newcommand{\complexMacro}{%
%    \end{macrocode}
% Initialize all of our counters to zero.
%    \begin{macrocode}
  \setcounter{count@i}{0}%
  \setcounter{count@ii}{0}%
  \setcounter{count@iii}{0}%
  \setcounter{count@iv}{0}%
%    \end{macrocode}
% \begin{macro}{\helperMacro}
% Define a helper macro for |\complexMacro| to use.
%    \begin{macrocode}
  \def\helperMacro#1,#2,\relax{%
```

```
    \someOtherMacro{#1}{#2}%
  }%
%    \end{macrocode}
% Do some really complicated processing.
%    \begin{macrocode}

                ⋮


%    \end{macrocode}
% We're all finished now.
%    \begin{macrocode}
}
%    \end{macrocode}
% \end{macro}
% \end{macro}
```

请注意，上述代码在\complexMacro内部定义了\helperMacro。文档使用了一个嵌套的\begin{macro}来引入 \helperMacro，按照惯例，在外层宏的\end{macro}之前放置了一个\end{macro}，而不是紧跟在\helperMacro的最后的 "}" 后面。

附录 A.3 列出了一个完整的、包含 .sty 文件及其文档的 .dtx 文件的框架。

类文件 从 .dtx文件生成类文件的过程比生成样式文件的过程复杂得多。问题在于 \DocInput依赖于 \usepackage{⟨*package*⟩}行（更准确地说，依赖于⟨*package*⟩.sty中的 \ProvidesPackage行）来设置 \fileversion和\filedate宏。然而，类文件无法使用\usepackage加载。我们也不能简单地用 \documentclass{⟨*package*⟩}加载，因为一个文档只能加载一个类，并且我们需要加载的类是 ltxdoc。

解决方法是使用 \ProvidesFile来使文件版本和日期可供 .dtx文件使用。附录 A.4 列出了一个完整的、包含 .cls 文件及其文档的 .dtx 文件的框架。它类似于附录 A.3 中展示的框架文件，但头部部分结构不同。

4 技巧、窍门和建议

• 多写好文档！这确实有助于其他人理解你的代码和整个包。

• 如果你认为整个 LaTeX社区都对你的包感兴趣，那么你应该将它上传到 CTAN，网址是http://www.ctan.org/upload。作为所有与 TeX 相关事物的中央存储库，CTAN会比你个人主页上的位置更容易让其他人找到你的 LaTeX包。

• 在分发你的包时，务必包含一个描述你的包功能的 README 文件，以及最好是作为 PDF 文件的预构建文档。预构建的文档能够让用户免去下载你的包、安装它以及在了解包的功能或是否符合他们需求之前构建文档的麻烦。

• 使用 LaTeX的分段命令来组织代码并澄清其结构（例如 \subsection{初始化宏}、\subsection{辅助函数}、\subsection{导出的宏和环境}，等等）。

• 虽然注释确实只属于排版文档，但也可以编写只在 .sty文件中可见的注释，以及在排版文档和 .sty文件中都可见的注释，或者只在 .dtx源文件中可见的注释。表 3 显示了如何控制注释的可见性。

Table 3: Comment visibility

Appears in docs	Appears in .sty	Mechanism	
N	N		% ^^A ⟨*comment*⟩
N	Y		% \iffalse %% ⟨*comment*⟩ % \fi
Y	N		% ⟨*comment*⟩
Y	Y		%% ⟨*comment*⟩

• 在 <*package> 和 </package> 之间的所有行（除了在 macrocode环境内的行），都应该以 “%” 开头。不要使用空行；这些空行会被写入到 .sty文件中（而且不应该）。

• 对于仅在包内部使用的宏、长度、计数器等，在其名称中使用 “@” 是个好习惯，可以声明为全局，但只打算在包内部使用。这样可以防止用户通过无意中重新定义包内部而破坏包状态。[6] 另一个好习惯是为包内部的所有全局名称添加包的名称前缀（例如 “\⟨*package*⟩@thing” 而不是 “\@thing” 或者更糟糕的是只是 “\thing”）。这有助于避免包之间的命名冲突。最后，因为十进制数字通常不允许在宏名称中，因此常常使用罗马数字，例如：\arg@i、\arg@ii、\arg@iii、\arg@iv等。

• 除了宏和环境之外，你可以像通常一样使用 \index索引其他内容。

• 因为宏名称可能很长，考虑使用 idxlayout包来减少索引中的列数。（它还提供了对索引格式的其他方面的控制。）

• 如果你使用 Emacs 作为文本编辑器，请尝试 swiftex.el 的doctex-mode，这是专门用于编写 .dtx 文件的 Emacs 模式。swiftex.el 可以从 CTAN获取。

作为更原始的替代方法，查阅 Emacs的 string-rectangle 和 kill-rectangle 命令。这些命令对于在区域内的每一行开头添加和删除 “%” 非常有帮助。

• 一定要阅读《DocStrip 程序》[7] 和《doc 和 shortvrb 宏包》[8] 的文档，分别是 DocStrip 和 Doc 的文档（当然是以 .dtx格式提供的）。这些文档解释了如何使用 .ins 和 .dtx 文件进行比本教程涵盖的更高级的操作。一些高级主题包括以下内容：

– 从单个 .dtx 文件中提取多个 .sty 文件。

– 将不同的导言放入不同的 .sty 文件中。

[6] 在 LaTeX 文档中，“@” 被设置为类别码 12（“其他”），而不是类别码 11（“字母”），所以用户不能轻易地定义或使用带有 “@” 的宏。

[7] 译者注：该文档可对照参阅由本人翻译的中文版。

[8] 译者注：该文档可对照参阅由本人翻译的中文版。

– 从 .dtx 文件中提取除了 .sty 文件之外的其他内容（例如配置文件或 Python 代码）。

– 更改排版文档的格式。

5 高级打包技术

本节介绍了使用 Doc 和 DocStrip 可以实现的各种高级技巧。虽然很少有包需要这些技术，但它们被包含在这里以供方便参考。

5.1 主文档文件

Doc支持包含多个 .dtx文件的“主”文档文件。其优点在于，可以使用连续的节编号和单一的统一索引对一组相关的 .dtx文件进行排版。事实上，LaTeX 2ε源代码本身就是使用一个主文档（source2e.tex）排版的，其中包含了构成 LaTeX 2ε的众多 .dtx文件。

为了帮助生成主文档，ltxdoc类提供了一个命令称为“\DocInclude”。ltxdoc的 \DocInclude与 Doc 的 \DocInput非常相似——它甚至在内部使用它——但具有以下附加功能。

- \PrintIndex会被自动正确处理。
- 每个 \DocInclude的文件都会有一个标题页。
- \tableofcontents按预期工作。.dtx文件名被用作“章节”名称。

注意，与 \DocInput不同，\DocInclude假定使用 .dtx扩展名。

附录 A.5 提供了一个主文档的框架，使用 \DocInclude将 ⟨*file1*⟩.dtx、⟨*file2*⟩.dtx 和 ⟨*file3*⟩.dtx排版为单个文档。如果你更喜欢手动操作（例如，如果你不喜欢 \DocInclude为每个文件生成标题页），你仍然可以使用 \DocInput。只需确保重新定义 \PrintIndex为空操作；否则，每个文件都会有自己的索引。在所有 .dtx文件都排版完之后，调用原始的 \PrintIndex命令以打印一个统一的索引：

```
\begin{document}
  \let\origPrintIndex=\PrintIndex \let\PrintIndex=\relax
  \DocInput{⟨file1⟩.dtx}
  \DocInput{⟨file2⟩.dtx}
  \DocInput{⟨file3⟩.dtx}
  \origPrintIndex
\end{document}
```

5.2 单文件包发布

尽管 LaTeX包通常以 .ins和 .dtx文件的形式进行分发，但也可以将包作为单个文件进行分发。关键在于在 .dtx文件的顶部，在%⟨*package*⟩行之后，包含整个 .ins：

```
%<*batchfile>
\begingroup
⋮
⟨Entire contents of the .ins file⟩
⋮
\endgroup
%</batchfile>
```

省略 \endbatchfile以允许 LaTeX继续处理 .dtx文件的其余部分。另外，为了避免出现“File ⟨*sty-file*⟩ already exists on the system. Overwrite it? [y/n]”的消息，你可以在第一个 \generate命令之前加上“\askforoverwritefalse”。（这将自动覆盖现有的 .sty文件。将 \generate命令放在“\IfFileExists{⟨*sty-file*⟩}{}{…}”中将会阻止覆盖。）你还应该将 .sty的安装说明移到 .dtx文件的末尾，这样用户就不会将其滚动到屏幕外。你需要使用 \typeout，因为 \Msg不会被定义：

```
% \Finale
%
```

```
% \typeout{**************************************************}
% \typeout{*}
% \typeout{* To finish the installation you have to move the}
% \typeout{* following file into a directory searched by TeX:}
% \typeout{*}
% \typeout{* \space\space skeleton.sty}
% \typeout{*}
% \typeout{* Documentation is in skeleton.dvi.}
% \typeout{*}
% \typeout{* Happy TeXing!}
% \typeout{**************************************************}
\endinput
```

5.3 具有共享版本信息的类和样式文件

某些包包含 .cls和 .sty文件。希望从同一个 .dtx文件中提取这两个文件，并共享相同的版本信息字符串。《DocStrip 程序》文档解释了如何从单个 \generate调用中提取多个文件：

```
\generate{\file{⟨package⟩.cls}{\from{⟨package⟩.dtx}{class}}
          \file{⟨package⟩.sty}{\from{⟨package⟩.dtx}{package}}}
```

可以通过更改附录 A.4中显示的 .dtx文件中的以下行来为 .cls和 .sty文件使用单个版本字符串：

```
%<class>\NeedsTeXFormat{LaTeX2e}[2023-11-01]
%<class>\ProvidesClass{⟨package⟩}
%<*class>
    [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨brief description⟩]
%</class>
```

替换的代码指定了哪些行属于类文件，哪些属于样式文件：

```
%<class|package>\NeedsTeXFormat{LaTeX2e}[2023-11-01]
%<class>\ProvidesClass{⟨package⟩}
%<package>\ProvidesPackage{⟨package⟩}
%<*class|package>
    [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨brief description⟩]
%</class|package>
```

5.4 高级打包技术示例集

在 CTAN https://www.ctan.org/tex-archive/info/dtxgallery上查看 .dtx 示例集，其中包括以下内容的示例：

- 单文件包发布（参见第 5.2 节）
- 条件代码包含（参见表 3 ）
- 重新排列代码以在文档中展示

A 框架文件

本节包含了文档其余部分讨论的文件类型的完整框架。这些框架可以作为创建自己的包的模板。

A.1 用于生成 .sty 文件的框架 .ins 文件

```
%%
%% Copyright (C) ⟨year⟩ ⟨your name⟩
%%
%% This file may be distributed and/or modified under the
%% conditions of the LaTeX Project Public License, either
%% version 1.3 of this license or (at your option) any later
%% version.  The latest version of this license is in:
%%
```

```
%%    http://www.latex-project.org/lppl.txt
%%
%% and version 1.3c or later is part of all distributions of
%% LaTeX version 2008-05-04 or later.
%%

\input docstrip.tex
\keepsilent

\usedir{tex/latex/⟨package⟩}

\preamble

This is a generated file.

Copyright (C) ⟨year⟩ ⟨your name⟩

This file may be distributed and/or modified under the
conditions of the LaTeX Project Public License, either
version 1.3 of this license or (at your option) any later
version.  The latest version of this license is in:

   http://www.latex-project.org/lppl.txt

and version 1.3c or later is part of all distributions of
LaTeX version 2008-05-04 or later.

\endpreamble

\generate{\file{⟨package⟩.sty}{\from{⟨package⟩.dtx}{package}}}

\Msg{************************************************************}
\Msg{*}
\Msg{* To finish the installation you have to move the}
\Msg{* following file into a directory searched by TeX:}
\Msg{*}
```

```
\Msg{* \space\space ⟨package⟩.sty}
\Msg{*}
\Msg{* To produce the documentation run the file ⟨package⟩.dtx}
\Msg{* through LaTeX.}
\Msg{*}
\Msg{* Happy TeXing!}
\Msg{*************************************************************}

\endbatchfile
```

A.2 用于生成 .cls 文件的 .ins 框架文件

```
%%
%% Copyright (C) ⟨year⟩ ⟨your name⟩
%%
%% This file may be distributed and/or modified under the
%% conditions of the LaTeX Project Public License, either
%% version 1.3 of this license or (at your option) any later
%% version.  The latest version of this license is in:
%%
%%    http://www.latex-project.org/lppl.txt
%%
%% and version 1.3c or later is part of all distributions of
%% LaTeX version 2008-05-04 or later.
%%

\input docstrip.tex
\keepsilent

\usedir{tex/latex/⟨class⟩}

\preamble

This is a generated file.
```

```
Copyright (C) ⟨year⟩ ⟨your name⟩

This file may be distributed and/or modified under the
conditions of the LaTeX Project Public License, either
version 1.3 of this license or (at your option) any later
version.  The latest version of this license is in:

   http://www.latex-project.org/lppl.txt

and version 1.3c or later is part of all distributions of
LaTeX version 2008-05-04 or later.

\endpreamble

\generate{\file{⟨class⟩.cls}{\from{⟨class⟩.dtx}{class}}}

\Msg{************************************************************}
\Msg{*}
\Msg{* To finish the installation you have to move the}
\Msg{* following file into a directory searched by TeX:}
\Msg{*}
\Msg{* \space\space ⟨class⟩.cls}
\Msg{*}
\Msg{* To produce the documentation run the file ⟨class⟩.dtx}
\Msg{* through LaTeX.}
\Msg{*}
\Msg{* Happy TeXing!}
\Msg{************************************************************}

\endbatchfile
```

A.3 用于生成 .sty 文件的 .dtx 框架文件

```
% \iffalse meta-comment
%
```

```
% Copyright (C) ⟨year⟩ ⟨your name⟩
% ---------------------------------
%
% This file may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in:
%
%    http://www.latex-project.org/lppl.txt
%
% and version 1.3c or later is part of all distributions of LaTeX
% version 2008-05-04 or later.
%
% \fi
%
% \iffalse
%<package>\NeedsTeXFormat{LaTeX2e}[2023-11-01]
%<package>\ProvidesPackage{⟨package⟩}
%<package>    [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨brief description⟩]
%
%<*driver>
\documentclass{ltxdoc}
\usepackage{⟨package⟩}
\EnableCrossrefs
\CodelineIndex
\RecordChanges
\begin{document}
  \DocInput{⟨package⟩.dtx}
\end{document}
%</driver>
% \fi
%
% \changes{v1.0}{⟨YYYY⟩-⟨MM⟩-⟨DD⟩}{Initial version}
%
% \GetFileInfo{⟨package⟩.sty}
%
```

```
% \DoNotIndex{⟨list of control sequences⟩}
%
% \title{The \textsf{⟨package⟩} package\thanks{This document
%   corresponds to \textsf{⟨package⟩}~\fileversion,
%   dated \filedate.}}
% \author{⟨your name⟩ \\ \texttt{⟨your e-mail address⟩}}
%
% \maketitle
%
% \begin{abstract}
%   Put text here.
% \end{abstract}
%
% \section{Introduction}
%
% Put text here.
%
% \section{Usage}
%
% \DescribeMacro{\YOURMACRO}
% Put description of macro |\YOURMACRO| here.
%
% \DescribeEnv{YOURENV}
% Put description of environment |YOURENV| here.
%
% \MaybeStop{\PrintIndex}
%
% \section{Implementation}
%
% \begin{macro}{\YOURMACRO}
% Put explanation of |\YOURMACRO|'s implementation here.
%    \begin{macrocode}
\newcommand{\YOURMACRO}{}
%    \end{macrocode}
% \end{macro}
%
```

```
% \begin{environment}{YOURENV}
% Put explanation of |YOURENV|'s implementation here.
%    \begin{macrocode}
\newenvironment{YOURENV}{}{}
%    \end{macrocode}
% \end{environment}
%
% \Finale
\endinput
```

A.4 用于生成 .cls 文件的 .dtx 框架文件

```
% \iffalse meta-comment
%
% Copyright (C) ⟨year⟩ ⟨your name⟩
% ------------------------------
%
% This file may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in:
%
%    http://www.latex-project.org/lppl.txt
%
% and version 1.3c or later is part of all distributions of LaTeX
% version 2008-05-04 or later.
%
% \fi
%
% \iffalse
%<*driver>
\ProvidesFile{⟨class⟩.dtx}
%</driver>
%<class>\NeedsTeXFormat{LaTeX2e}[2023-11-01]
%<class>\ProvidesClass{⟨class⟩}
```

```
%<*class>
    [⟨YYYY⟩-⟨MM⟩-⟨DD⟩ v⟨version⟩ ⟨brief description⟩]
%</class>
%
%<*driver>
\documentclass{ltxdoc}
\EnableCrossrefs
\CodelineIndex
\RecordChanges
\begin{document}
  \DocInput{⟨class⟩.dtx}
\end{document}
%</driver>
% \fi
%
% \changes{v1.0}{⟨YYYY⟩-⟨MM⟩-⟨DD⟩}{Initial version}
%
% \GetFileInfo{⟨class⟩.dtx}
%
% \DoNotIndex{⟨list of control sequences⟩}
%
% \title{The \textsf{⟨class⟩} class\thanks{This document
%   corresponds to \textsf{⟨class⟩}~\fileversion,
%   dated \filedate.}}
% \author{⟨your name⟩ \\ \texttt{⟨your e-mail address⟩}}
%
% \maketitle
%
% \begin{abstract}
%   Put text here.
% \end{abstract}
%
% \section{Introduction}
%
% Put text here.
%
```

```
% \section{Usage}
%
% \DescribeMacro{\YOURMACRO}
% Put description of macro |\YOURMACRO| here.
%
% \DescribeEnv{YOURENV}
% Put description of environment |YOURENV| here.
%
% \MaybeStop{\PrintIndex}
%
% \section{Implementation}
%
% \begin{macro}{\YOURMACRO}
% Put explanation of |\YOURMACRO|'s implementation here.
%    \begin{macrocode}
\newcommand{\YOURMACRO}{}
%    \end{macrocode}
% \end{macro}
%
% \begin{environment}{YOURENV}
% Put explanation of |YOURENV|'s implementation here.
%    \begin{macrocode}
\newenvironment{YOURENV}{}{}
%    \end{macrocode}
% \end{environment}
%
% \Finale
\endinput
```

A.5 主文档框架文件（.tex）

```
\documentclass{ltxdoc}
\usepackage{⟨file1⟩}
\usepackage{⟨file2⟩}
\usepackage{⟨file3⟩}
```

```
\title{⟨title⟩}
\author{⟨you⟩}

\EnableCrossrefs
\CodelineIndex
\RecordChanges

\begin{document}
  \maketitle

  \begin{abstract}
    Put text here.
  \end{abstract}

  \tableofcontents

  \DocInclude{⟨file1⟩}
  \DocInclude{⟨file2⟩}
  \DocInclude{⟨file3⟩}
\end{document}
```

参考文献

[1] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison Wesley, Reading, Massachusetts, October 1, 1994. ISBN 0-201-54199-8.

[2] Donald E. Knuth. Literate programming. *The Computer Journal*, 27(2):97–111, May 1984. British Computer Society. Available from http://www.literateprogramming.com/knuthweb.pdf.

[3] LaTeX Project Team. LaTeX for package and class authors. Available from https://ctan.org/pkg/clsguide, October 24, 2023.

索引