PUTEX 4.0 GBK 版使用手册 (Rev 1.0)

静宜大学信息管理学系
蔡奇伟
cwtsay@pu.edu.tw
http://www.cs.pu.edu.tw/~tsay/putex *

September 15, 2004

*本计画部份是由国科会提供经费赞助完成（计画编号：NSC-86-2213-E-126-005）。

说明

由于笔者没有简体中文系统与字型，所以本文件编排的效果非常简陋。安装完 pTEX 之后，请用 gblatex 来重新编排本文件的原始档：guide40_gb.tex。本文件使用了宋体、中黑、仿宋、与隶书等字型。若你有这些字型，就可以产生和 Big5 版同样美观的文件。

目录

1 前言

这份手册说明如何使用 PUTEX 4.0。笔者假设你曾使用过 TEX/LATEX 来排版文件，对 TEX/LATEX 已有基本的知识，所以本手册不会论及 TEX/LATEX 的使用明细与内部原理，比方说：本文件不会告诉你如何产生 α, β, ... 等的希腊字母，也不会教你如何写出类似 $\sqrt{x^2+1}$ 的数学公式，更不会谈到 glue, box, line breaking 之类的概念。所以，如果你是一位 TEX 新手，笔者建议你先阅读一些 TEX 或 LATEX 的书籍（如参考书目中的 [2, 3, 4, 5]），等具备了一些基本观念后，再回来细读此手册。

PUTEX 构筑在 Christian Schenk 先生的 MikTEX 系统上。 PUTEX 本身只包含修改过的 TEX 程序与几个辅助程序，其他如 TEX 字体、 LATEX 样式定义档、 DVI 驱动程序等，仍然得仰赖 MikTEX 系统的支持。因此安装 PUTEX 之前，你必须先在计算机中先安装好 MikTEX。

为什么取名为 PUTEX 呢？ 这是因为笔者服务于静宜大学，PU 两个英文字母是静宜大学英文校名：Providence University 的字头。因此，PUTEX 应该读成 P·U·TEX。

1.1 如何取得 PUTEX?

PUTEX 计画的 WWW 主页是：

`http://www.cs.pu.edu.tw/~tsay/putex/`

你可以从以上网址下载最新版的 PUTEX 软件、本手册的最新版本、与浏览 TEX 相关的信息。由于 PUTEX 必须在 MikTEX 的环境下才能运行，若你尚未安装 MikTEX 的话，也请从上述的网站一并取得映射的 MikTEX 系统。

1.2 PUTEX 的特点

PUTEX 是一套在 Windows 系统下运行的中文 TEX 排版系统。 PUTEX 改写 D. E. Knuth 教授 TEX 程序的原始码，使之能够直接处理汉字元。由于不是采用前置处理（preprocessing）的方法来处理 TEX 文件中的汉字元， PUTEX 比其他的中文 TEX 系统具有更高度的方便性、扩充性、与弹性。此外，PUTEX 还具有以下的功能：

- 支持所有厂牌（如华康、文鼎、全真、或新研泽）、所有字貌的 TrueType 汉字型（如细明、楷书、行书、或勘亭流等）。

  注：PUTEX 并不直接支持英文 TrueType 字体。

- 可使用 TrueType 中文外字符集字体。

- 具有标点符号避行首 / 行尾的排版功能，如〝。〞和〝，〞等标点符号将不出现在行首；而〝（〞和〝【〞等标点符号也不出现在行尾。

- 支持以中文来命名宏命令。
- 可自由调整汉字的字间距（character spacing）。
- 可自由调整汉字与英文字之间的字间距。
- 可自由调整汉字基线（baseline）的位置。
- 支持中文直排的功能。
- 支持以中文数字表示法来显示整数值的功能，让你能够轻松地制作出以〝一、二、三、...〞或以〝壹、贰、参、...〞为标号的条列项目。
- 产生具有可携性的 CDI（CJK Device Independent file）文件。
- 提供 CDI 至 DVI（DeVice Independent file）的转档程序，所产生的 DVI 档可以用 MikTEX 的 Yap 来打印预览、用 dvips 转换成 PostScript 档、或用 dvipdfm 转换成 PDF 档。
- 提供 CDI 至 PDF 的转档程序。
- 提供制作中英文索引的 puidx 程序（注：当前只支持 Big5 码）。

在底下的各节中，我们会进一步地阐明如何使用这些功能。

1.3 PUTEX 的软、硬件需求

PUTEX 的软、硬件最低需求如下：

软件： Windows 操作系统与 MikTEX。TrueType 汉字型可依个人所需安装。笔者建议你最少应有细明、楷书、和中黑这三种常用的字体。

硬件： Pentium 90Mz CPU, 16 MB RAM, 500 MB 以上的硬盘空间。

另外，你最好使用 15 寸（或以上）的显示器，因为你需要至少 800×600 的屏幕分辨率，才能够比较清晰地预览 DVI 或 CDI 档。

1.4 安装 PUTEX

请以 WWW 浏览器阅读文件 `install.html`，并按照其中所述的步骤来进行安装。安装 PUTEX 之后，请务必依照附录 A 所述的方式来设置 `cfont.map` 这个字体映射档，让 PUTEX 能够正确地利用 Windows 系统中的 TrueType 汉字型。

2 PUTEX 快速入门

PUTEX 的运行方式与英文版的 TEX 系统大致相同，即分为以下三个步骤：

一、 编辑 TEX 或 LATEX 格式的中文档档，如 `foo.tex`。

二、 选择下列适当的命令来产生 CDI 输出档：

- 如果 `foo.tex` 是 TEX 格式，则运行命令 `gbtex foo`
- 如果 `foo.tex` 是 LATEX 格式，则运行命令 `gblatex foo`

三、 选择下列的方式之一来预览或打印 CDI 档：

- 用 cdi2dvi 程序把 CDI 档转为 DVI 档，然后用 Yap 来预览或打印。
- 把前述转换所得的 DVI 档，用 dvips 转成 PostScript 档然后用 gsview 来预览或打印。
- 用 cdi2pdf 程序把 CDI 档转为 PDF 档，然后用 Adobe 公司的 Acrobat Reader 来预览或打印。

以下我们就这三个步骤，做进一步的说明：

2.1 编辑中文 TEX/LATEX 文件

编写包括中文的 TEX/LATEX 文件时，请注意以下几点：

- 汉字元不可以直接键入数学式中，而必须含括在 `\mbox`（或 `\hbox`）之中。比方说：

  ```
  $x^2 > 0 \mbox{ 若 } x \ne 0$
  ```

  产生的结果为：$x^2 > 0$ 若 $x \ne 0$。但是以下的输入方式：

  ```
  $x^2 > 0 若 x \ne 0$
  ```

  则会产生错误信息，而且其结果为：$x^2 > 0x \ne 0$（其中的〝若〞字消失不见）。

- 你可以用和英文 TEX 文件一样的方式撰写中文 TEX 文件。

- 撰写中文 LATEX 文件时，若想使用中文化的标题，请使用第 4 节所述的中文化文件类别（class）：`gbkarticle`, `gbkreport`, 或 `gbkbook`。比方说，本文件使用 `gbkarticle`，因此第一行是：

  ```
  \documentclass[11pt,a4paper]{gbkarticle}
  ```

- 若撰写以英文为主、中文只占少部分的文件（譬如附中文摘要的英文文章），直接使用 LATEX 的文件类别即可，譬如：

  ```
  \documentclass[11pt,a4paper]{article}
  ```

2.2 编译中文 TEX/LATEX 文件

假定 foo.tex 是一个 TEX 文件档。你必须在"命令提示字符"窗口中（或 Windows 9x 的 DOS 窗口中）键入命令：

```
gbtex foo
```
（若 foo.tex 是 TEX 格式）

或

```
gblatex foo
```
（若 foo.tex 是 LATEX 格式）

来进行排版。如果一切正确，上述的命令会在当前的文件夹中产生名为 foo.cdi 输出档。

由于 CDI 是一套扩充 DVI 格式的文件规格，其中包含中文码与汉字体等信息，所以常规的 DVI 驱动程序无法解读 CDI 档。你必须用底下三节所介绍的方式来预览或打印 CDI 档。

假定 cdi2dvi 已配置完成，你只要键入以下两行命令：

```
cdi2dvi foo
yap foo
```

即可在 Yap 的窗口中预览到排版的结果。由于 Yap 具有自动更新的功能，在屏幕上保留 Yap 窗口，之后只要键入

```
cdi2dvi foo
```

无须再运行 Yap 程序，即可在 Yap 窗口中看到更新的结果。

cdi2dvi 运行时，会在工作目录中新建一个名为 cdifont 的子目录。然后将产生的汉字型档与 tfm (tex font metric) 档置于其中。cdi2dvi 每次运行都会先删除旧的 cdifont 子目录，因此你不必担心汉字型档会日益增加，而占据了大量硬盘空间。当你完稿后，可用底下的命令自行删除 cdifont 目录

```
del/s/q cdifont
```
（Windows NT/2000/XP）

或

```
deltree cdifont
```
（Windows 95/98）

cdi2dvi 命令的语法如下：

cdi2dvi [*flags*] *cdi_file*[.cdi] [*dvi_file*[.dvi]]

其中，仅 *cdi_file* 不可缺外，其余的参数与扩展名均可省略。若没有指定 *dvi_file* 输出档，则 DVI 输出档将以 *cdi_file*.dvi 为名。cdi2dvi 的命令选项如下：

-p *mode* *x*_resolution *y*_resolution

设置打印机的输出模式名称（*mode*）、水平分辨率（*x*_resolution）、与垂直分辨率（*y*_resolution）。

-s *mode* *x*_resolution *y*_resolution

设置屏幕的输出模式名称（*mode*）、水平分辨率（*x*_resolution）、与垂直分辨率（*y*_resolution）。

常规而言，你只要按照安装手册 [6] 的步骤设置好 cdi2dvi 之后，就不需要使用上述的命令选项。

2.3 转换成 PostScript 档

用 cdi2dvi 转换所得的 DVI 档可以再用 dvips 转成 PostScript 格式。譬如：以下两行命令即可把 foo.cdi 转成 foo.ps：

```
cdi2dvi foo
dvips foo
```

2.4 使用 cdi2pdf 转档程序

cdi2dvi 产生的 DVI 档，虽然可以用 dvipdfm 程序转成 PDF 档，但是在所得的 PDF 档中，双字节的中文码被换成单词节的内码，而且内嵌的汉字型是 bitmap 格式。

cdi2pdf 工具程序是笔者修改 dvipdfmx[1] 程序的成果。 cdi2pdf 把 CDI 档直接换转成 PDF 档，并具有以下两项优点：

- 保留原来的中文码，因而在 Acrobat Reader 中可以使用文句搜索的功能。
- 内嵌 TrueType 汉字型，在屏幕上浏览更美观，而且字体数据只存使用到的字符，而不是整个字体文件，可有效地降低文件大小。

按照安装手册 [6] 的步骤设置好 cdi2pdf 之后，只要输入命令：

```
cdi2pdf foo
```

就可以把 `foo.cdi` 转成 `foo.pdf`。

[1] dvipdfm 的加强版，可处理 CJK TrueType 字体

注：由于 Acrobat Reader 6 会把打开的 PDF 档设为防写，所以运行 cdi2pdf 之前，你必须先关闭预览中的 PDF 档，才不会造成无法写入文件的程序错误。

注：当前版本的 cdi2pdf 不支持斜体字和反白字等字体的设置。此外，由于 PDF 的规格限制：(1) 外字符集字符可能无法正确地显示 (2) 文件中的图档格式只能使用 JPG, PNG, 或 GIF，并不支持 EPS。

3 字符间距

这一节介绍 pTEX 设置字符间距的策略，以及利用双字缩合（kerning）的技巧来排版出更美观的文件。

3.1 空白字符的处理

空白字符在 TEX 文件中具有语法与语意两种用途。在语法方面，空白字符用来分隔英文字（words）与分隔语元（tokens）。在语意方面，空白字符用来产生弹性宽度的空白间隔，并作为可能的断行点（line-breaking point）。底下两点是 TEX 处理空白字符的基本策略：

- 除非空白字符被定义成产生空白宽度的命令（如在 `\verb` 命令或 `verbatim` 环境中），否则，连续多个空白字符会被删减成一个空白字符。
- 跟随在宏命令之后的空白字符是用来隔断命令名称，排版时将全数删除，而不会产生空白宽度。

pTEX 允许用户自由调整中文的字间距和中英文的字间距。然而，这个功能增加了空白字符的处理复杂性，因为空白字符不再单纯地只产生英文的字间距，而可能产生以下三种空白间隔：

- 英文空白：英文字之间的空白间隔
- 中文空白：汉字之间的空白间隔
- 中英空白：汉字与英文字之间的空白间隔

绝大部分的情况下，pTEX 程序会正确地判断出空白的种类。底下我们列出一些关于空白的规则，让你能更精确地掌握空白字符的使用：

★ 汉字元之间不需要以空白字符隔开，pTEX 会自动在两者之间插入中文空白。此外，在汉字元间添加一个或一百个空白字符并无差异，因为它们所产生的排版效果与不添加空白字符完全相同。如果想增加某两个汉字元的间距，你可以使用 `\␣`、`\,`、或 `\hspace` 之类的命令。

★ 汉字元与英文字符之间不需要以空白字符隔开，pTEX 会自动在两者之间插入中英空白。此外，在两者之间添加一个或一百个空白字符并无差异，因为它们所产生的排版效果与不添加空白字符完全相同。

★ 汉字元与行间（inline）数学式之间不需要以空白字符隔开，pTEX 会自动在两者之间插入中英空白。此外，在两者之间添加一个或一百个空白字符并无差异，因为它们所产生的排版效果与不添加空白字符完全相同。譬如底下两行：

```
公式 $x^2=4$ 的正解为 $x=2$。
公式␣$x^2=4$␣的正解为␣$x=2$。
```

（符号 ␣ 代表空白字符）产生的结果均为：

公式 $x^2 = 4$ 的正解为 $x = 2$。

★ pTEX 把正常汉字元的类别码默认为 letter（参见 5.7 节），因此你必须用至少一个空白字符将 TEX 命令与其后的汉字元隔开，否则该汉字元会被视为命令名称之一部份，而造成错误。比方说：

```
我爱 \TeX␣的排版能力
```

产生的结果为

我爱 TEX 的排版能力

但是

```
我爱 \TeX 的排版能力
```

则会让 pTEX 误认〝 \TeX 的排版能力〞为一个命令，而造成指令未定义的错误。

★ pTEX 将中文符号字符的类别码默认为 other，因此 TEX 命令之后跟著中文符号时，你可以省略两者间的空白。比方说以下三种输入方式：

```
〝我爱 \TeX。〞 、〝我爱 \TeX␣。〞 、〝我爱 \TeX\␣。〞
```

都会产生相同的结果：〝我爱 TEX。〞然而，笔者建议你：

不要在中文标点符号之前使用空白命令 \␣。

否则会使得该标点符号的〝避首 / 尾〞功能失效（参见第 6.2 节）。

★ 请使用空白字符将 TEX/LATEX 命令与前后文字隔开，让 pTEX 来决定这些空白的种类。

★ 若命令前后的空白字符无法产生正确的空白间距时，你可以将其改为 \␣。通常这样就可以解决大部分的问题。

★ 若改为 \␣ 后仍无法获得正确的空白，你可以用 pUTEX 的命令 \PUXspace（英文空白）、\PUXcspace（中文空白）、或 \PUXcespace（中英空白）来直接设置所需的空白。

★ 你可以在两个字符间插入 \kern0pt 或 \hbox{} 来制止 pUTEX 在两者之间添加空白间距。譬如：底下三行的输入方式：

```
我喜欢在 WWW 上搜索数据
我\hbox{}喜\hbox{}欢\hbox{}在\hbox{}WWW\hbox{}上搜索数据
我\kern0pt喜\kern0pt欢\kern0pt在\kern0ptWWW\kern0pt上搜索数据
```

所产生的结果分别为：

我喜欢在 WWW 上搜索数据
我喜欢在WWW上搜索数据
我喜欢在WWW上搜索数据

第二、三行的长度明显地短于第一行的长度，因为字符间并未添加空白间距。又比方说，\CDOTS 命令（定义在 chinesebasic 档中）可用来产生中式连点“……”，它的定义如下：

```
\newcommand*{\CDOTS}{\mbox{...\kern0pt...}}
```

如果两个... 之间不加 \hbox{} 的话，就会得到“……”中间多出一个中文空白的宽度。

★ 在 \verb 命令与 verbatim 环境中的任何空白字符将会保留而不删除，并产生英文空白字符的宽度。譬如：

```
\verb|姓名␣␣␣年龄␣␣␣住址␣␣␣Phone|
```

产生的结果为：

```
姓名   年龄   住址   Phone
```

列完以上的通则后，底下我们举一些输入方式的范例。

例 –1 如果宏命令展开后的第一个语元是中文或英文字符（如 \TeX），则你不必在该宏命令之前键入空白字符。譬如：假定命令 \A 的定义为：

```
\newcommand{\A}{排版}
```

则底下两行输入方式：

```
使用␣\TeX␣来␣\A␣真方便！
使用\TeX␣来\A␣真方便！
```

会产生相同的结果如下：

使用 TEX 来排版真方便！
使用 TEX 来排版真方便！

例 –2 如果你使用字体设置群组，如 `{\it ...}` 或 `{\itshape ...}` 等等，则你不必在其群组之前后键入空白字符。譬如底下的两行输入方式：

```
\PUTeX␣处理空白的␣{\it 规则一 }␣是 ...
\PUTeX␣处理空白的 {\it 规则一 } 是 ...
```

会产生相同的结果如下：

PUTEX 处理空白的规则一是 ...
PUTEX 处理空白的规则一是 ...

在少数的状况下，PUTEX 还是得要靠你的辅助才能正确地处理字符间的空白间隔。底下，我们示范用 `\␣` 来替换空白字符 ␣ 以解决错误空白间隔的问题。

例 –3 除了〝 `\verb` 命令中的第一个字符是汉字元〞这个情况外， `\verb` 命令前后的空白字符都会产生正确的空白间隔。当 `\verb` 命令中的第一个字符是汉字元时，你必须用 `\␣` 来替换 `\verb` 命令之前的空白字符 ␣。譬如：

```
我服务于␣\verb|Providence University|␣资管系。
我服务于\␣\verb| 静宜大学 |␣资管系。
我服务于␣\verb| 静宜大学 |␣资管系。
```

产生的结果分别为：

我服务于 `Providence University` 资管系。
我服务于静宜大学资管系。
我服务于 静宜大学资管系。

前两行的结果是正确的；最后一行〝 于〞与〝 静〞之间被误认成英文空白，因此显得稍宽而不美观。

例 –4 由于 `\underline` 命令是利用数学式的技巧来制作，因此 PUTEX 会将下划线字与前后中文之间的空白视为中英空白。譬如：

```
你␣\underline{ 千万不可以 }␣说谎
```

产生的结果为：

你 千万不可以 说谎

句中下划线字与前后汉字的间距显然不对。这时你可以用 `\␣` 来替换空白字符 ␣：

你 \␣\underline{ 千万不可以 }\␣ 说谎

而得到底下的正确结果：

你<u>千万不可以</u>说谎

例 3–5 假定下划线字的首尾是英文字符，则其前后的空白会正确地解释成英式空白。譬如：

你 ␣\underline{never}␣ 说谎

将产生正确的结果：

你 <u>never</u> 说谎

例 3–6 有些命令（如 \index）会在前后〝暗中〞添加一些 TEX 内部数据结构，使得 PUTEX 无法正确地判断空白的种类，即使用前述之 \␣ 的技巧也无法解决。比方说，在本手册中，笔者定义如下的命令：

```
\newcommand*{\MikTeX}{Mik\TeX\index{MikTeX}}
```

让命令\MikTeX 除了在本文中显示出 MikTEX 以外，也自动地添加索引之中。以下三种输入方式：

仍然得仰赖 \MikTeX␣系统的支持。
仍然得仰赖 \MikTeX\␣系统的支持。
仍然得仰赖 \MikTeX{}␣系统的支持。

产生的结果如下：

仍然得仰赖 MikTEX系统的支持。
仍然得仰赖 MikTEX系统的支持。
仍然得仰赖 MikTEX 系统的支持。

第一种输入方式：前者 X 之后的空白被 TEX 吸收掉，使得两字之间并无空白间隔，也使得 PUTEX 误添加中文空白。第二种输入方式： \␣ 被 PUTEX 解释成中式空白。第三种则产生正确的结果，其中的 {}（括弧之间不可有空白字符）阻止 TEX 吸收掉其后的空白字符。

例 3–7 你可以定义宏命令来避免多打空白字符。譬如有了以下的定义：

```
\newcommand{\pgref}[1]{ 第 \pageref{#1}␣页 }
```

你就可以打

```
请参阅 \pgref{...} 之说明                %（〝之〞字前不需空白字符）
```

来替换

```
请参阅第 \pageref{...}␣页之说明      %（〝页〞字前需要空白字符）
```

3.2 双字缩合

某些英文字符并置在一起时，由于字形的关系，会显得间距过宽，譬如底下左边所示的 A 和 V 两个字母：

AV AV

双字缩合（kerning）是把两个字符缩近距离，以得到更美观的效果，如上面右边所示的 A 和 V 两个字母。TEX 之所以是一个出色的排版软件，其中的一个原因是：不需要使者者操心，它就能自动地处理英文字间的双字缩合。

正常的汉字元都是方块字，不太需要做双字缩合，但是中文标点符号则不然，譬如底下的句子：

（“ 之”字前不需空白字符），如此即完成。

前头的 “ 和之字的间距有点宽，后头的 ）和，之间显然过宽。你可以用 TEX 的 \kern 命令来缩短它们的宽度，譬如用以下的输入：

（“ \kern-0.25em 之”字前不需空白字符）\kern-0.5em，如此即完成。

（em 是长度单位，约等于当前字体的大小），即可产生底下比较均匀美观的结果：

（“之”字前不需空白字符），如此即完成。

不过，若每次都要如此繁琐地输入 \kern 命令，显然费时费力又容易出错。你可以在文件前端先定义好以下的中文标点符号命令：

```
\def\，{\kern -0.5em，}   % 把逗点往前缩 0.5em
\def\“{“\kern -0.25em}  % 把左引号往前缩 0.25em
```

当需要缩合时，就用这些命令来替换原来的标点。譬如底下我们用 \“ 来替换左引号和用\，来替换逗点：

（\“ 之”字前不需空白字符）\，如此即完成。

同样地可以得到前述较美观的结果。

由于 PUTEX 把中文标点符号的类别默认为 other，所以前述的中文标点符号命令之后不需要以空白来隔断，这让你可以更轻松地使用它们。

笔者撰写本文时，即在文件前端定义了若干中文标点符号命令，用在所有需要缩合的地方。有兴趣的读者可以查看本文原始档，看看它们的定义方式与使用时机。如果读者手边还保留有旧版的 PUTEX 手册，拿来和本手册相对照，即可以发现新手册标点符号的编排比较紧凑美观。

注：由于中文标点符号命令指定的缩合距离与使用的汉字貌有关，没有放诸四海皆准的数值，因此 pTeX 不把它们定为标准命令，而必须由用户自行定义之。在笔者构想出高效率的中文双字缩合完整解决方案之前，你可以用前述的技巧来达成双字缩合的需求。

4 LaTeX 中文化

为了让用户能够更方便地利用 LaTeX 来产生中文档， gbLaTeX 提供以下三种文件类别（document class）：

`gbkarticle`	中文化的 LaTeX article 文件类别。
`gbkreport`	中文化的 LaTeX report 文件类别。
`gbkbook`	中文化的 LaTeX book 文件类别。

此外，针对某些与 gbLaTeX 不兼容的套件（如 `fancyvrb`），gbLaTeX 提供 `twpkgpatch` 套件来解决不兼容的问题。

4.1 chinesebasic 套件

这个套件包含中文化的共同设置，而且自动被前述的中文化文件类别加载。因此，以下所述的宏命令，同样适用于中文化文件类别。

pTeX 的 Logo

`chinesebasic` 定义了以下三个 Logo 命令：

- `\PUTeX` 产生 pTeX。
- `\PULaTeX` 产生 pLaTeX。
- `\PULaTeXe` 产生 pLaTeX 2_ε。

中式连点

`chinesebasic` 套件提供 `\CDOTS` 命令，其定义如下：

```
\newcommand*{\CDOTS}{\mbox{...\kern0pt...}}
```

比方说，若你输入：

鼠、牛、虎、兔、`\CDOTS`、狗、猪

所得之结果将为：

鼠、牛、虎、兔、……、狗、猪

注：请勿输入连续两个〝……〞汉字元来产生上述的中式连点，理由如第 13 页所述。

以中文数字显示 **counter** 的值

chinesebasic 套件提供的 \cnumber 或 \Cnumber 命令可用来将 counter 的数值以小写或大写中文数字的方式显示。它们的作用有如 LaTeX 的 \arabic 命令。譬如：

```
\cnumber{\thesection}     % 以小写中文数字显示 counter \thesection 的值
\Cnumber{\thesection}     % 以大写中文数字显示 counter \thesection 的值
```

中式编号条列

chinesebasic 套件提供了两个类似 enumerate 的中式编号条列环境。一个是以小写中文数字来编号，另一个则是以大写中文数字来编号。我们用以下的例子来说明它们的用法与所得的结果。

例 –1 底下是使用〝一二三〞小写中文数字编号条列环境的范例。左侧是输入方式，右侧则为排版的结果。

输入行

```
\begin{一二三}
  \item 首先，
    \begin{一二三}
      \item 第一条
        \begin{一二三}
          \item 第一项
            \begin{一二三}
              \item 第一点
            \end{一二三}
        \end{一二三}
    \end{一二三}
  \item 其次，
  \item 最后，
\end{一二三}
```

排版结果

一、 首先，

1. 第一条

(a) 第一项

i. 第一点

二、 其次，

三、 最后，

例 –2 底下是使用〝壹贰参〞大写中文数字编号条列环境的范例。左侧是输入方式，右侧则为排版的结果。

输入行	排版结果
\begin{壹贰参} \item 首先， \begin{壹贰参} \item 第一条 \begin{壹贰参} \item 第一项 \begin{壹贰参} \item 第一点 \end{壹贰参} \end{壹贰参} \end{壹贰参} \item 其次， \item 最后， \end{壹贰参}	壹、 首先， 1. 第一条 (a) 第一项 i. 第一点 贰、 其次， 参、 最后，

4.2 标题中文化

中文化的文件类别：gbkarticle、gbkreport、和gbkbook会把英文标题改成中文。表格1列出所更改的标题。若你不满意任何一个中文标题的名称，你可以用 \renewcommand 命令重新定义之。譬如，若想将默认的中文标题“序”改成“自序”，你只要在文件前端键入如下的一行即可：

```
\renewcommand*{\prefacename}{自序}  % Preface
```

4.3 gbkarticle 文件类别

gbkarticle.cls 是 LaTeX article.cls 的中文化文件类别。你可以用下面的方式来使用它：

```
\documentclass[options]{gbkarticle}
```

gbkarticle.cls 与 article.cls 功能相同，另外也加进了 chinesebasic 套件的内容。

4.4 gbkreport 与 gbkbook 文件类别

gbkreport 和 gbkbook 分别是 LaTeX report 和 book 的中文化文件类别。你可以用下面的方式来使用它们：

```
\documentclass[options]{gbkreport}
```

或

标题命令名称	原英文标题	中文标题	附注
\prefacename	Preface	序	
\partname	Part	篇	
\contentsname	Contents	目录	
\listfigurename	List of Figures	图形目录	
\listtablename	List of Tables	表格目录	
\indexname	Index	索引	
\appendixname	Appendix	附录	
\abstractname	Abstract	摘要	
\tablename	Table	表格	
\figurename	Figure	图	
\pagename	Page	页	
\refname	References	参考文献	article
\bibname	Bibliography	参考文献	report & book

表格 1: 中文标题

\documentclass[*options*]{gbkbook}

由于 gbkreport 和 gbkbook 雷同，因此我们就以 gbkreport 为例来说明两者的用法。

gbkreport 除了拥有 report 所有的功能以外，另外加进了 chinesebasic 套件的内容与中文标题的设置。此外，

- gbkreport 修改了 \chapter 命令与 \appendix 命令，使前者产生〝第一章〞之类的章序列号，后者产生〝附录 A〞之类的附录序列号。
- 目录里阿拉伯数字格式的章序列号也会改成〝第一章〞、〝第二章〞、之类的中式序列号。
- gbkreport 定义了 \twchaptername 宏命令，其内容是当前的中文章序列号，如〝第一章〞等等。

4.4.1 文件类别的选项

常规而言， gbkreport 的章序列号是使用小写中文数字，章标题则是左对齐。不过，你可以用以下的文件类别选项来改变这些设置：

chapscnum 章序列号使用俗体中文数字，如〝第廿一章〞。
chapucnum 章序列号使用大写中文数字，如〝第贰拾壹章〞。
chapfcnum 章序列号使用正式中文数字，如〝第壹拾壹章〞。

chapacnum	章序列号使用全角阿拉伯中文数字，如〝第２１章〞。
rightchhd	章标题右对齐。
centerchhd	章标题居中对齐。

比方说，以下的文件类别选项设置：

```
\documentclass[chapucnum,rightchhd]{gbkreport}
```

会使得章序列号改用大写中文数字，如〝第贰拾壹章〞，以及使得章标题右对齐。

注：目录及页首中的章序列号也会使用所设置的章序列号中文数字格式。

4.4.2 改变章序列号方式

如果想让 \chapter 产生〝第一讲〞，而非原本的〝第一章〞之类的章序列号，你只要重新定义命令 \chaptername 即可：

```
\renewcommand*{\chaptername}{讲}
```

4.4.3 改变章标题的字体

gbkreport 内部利用命令 \TWchapHeadingFont 和 \TWchapContentFont 来分别设置章标题及其在目录中所使用的字体。它们默认的定义如下：

```
\newcommand*{\TWchapHeadingFont}{\normalfont\Huge\bfseries}
\newcommand*{\TWchapContentFont}{\large\bfseries}
```

使得章标题里的中文在两个地方都使用中黑体字体。如果你想改变章标题的汉字型，可以重新定义以上的两个命令。譬如：

```
\renewcommand*{\TWchapHeadingFont}{\normalfont\Huge\bfseries\PUXFbr}
```

将使得章标题的中文改用粗圆体字体。同样地，以下的定义：

```
\newcommand*{\TWchapContentFont}{\Large\bfseries\PUXFmk}
```

将使得在目录里的章标题使用比较大的字体，以及中文改用中楷体字体。

注：如果你对以上的 \PUXFbr 与 \PUXFmk 命令感到〝雾煞煞〞，没事，因为它们是 PUTEX 用来更改汉字貌的命令。我们会在 5.2 节介绍它们。

4.4.4 中文引用号码

gbkreport 提供命令 \twref 让你产生中文引用号码。比方说，你可以用 \label 命令来定义某一章的标签，如：

```
\chapter{制作中文索引}\label{chap:makeidx}
```

然后利用以下的命令：

```
\newcommand{\chapref}[1]{第 \twref{chap:#1}\chaptername}
```

使得 \chapref{makeidx} 产生出如〝第八章〞而非〝第 8 章〞的中文引用号码。

由于命令 \twref 只是将引用号码的第一个数字改成中文数字，所以应用在节或数学公式时，会得到〝八 .1 节〞或〝公式九 .3〞的结果。基于这个限制，命令 \twref 比较适用于章号或页码等只由一个数字所组成的引用号码。

4.5 twpkgpatch 套件

gbLaTeX 只修改了 LaTeX 的 \verb 命令与 verbatim 环境的定义，使其中的空白字符能够正确地解读，gbLaTeX 其他所有的 LaTeX 命令都忠实地保留下来。因此 gbLaTeX 与 LaTeX 兼容度极高，几乎所有的 LaTeX 的套件都可以一成不变地使用在 gbLaTeX 之中。然而，有些套件提供类似 \verb 命令与 verbatim 的环境（如 fancyvrb ）就会与不兼容。因此 gbLaTeX 提供 twpkgpatch 套件来解决这类的问题。

以 fancyvrb 套件为例，你可以用下面的方式来解决不兼容的问题：

```
\usepackage{fancyvrb}
\usepackage[fancyvrb]{twpkgpatch}
```

即加载 fancyvrb 套件之后，再以 fancyvrb 为选项加载 twpkgpatch 套件。

注：到当前为止，笔者只发现 fancyvrb 套件与 gbLaTeX 不兼容。如果你发现到其他不兼容的套件时，请通知笔者，我会尽力加以解决。这种状况应该是极少发生的吧：）

4.6 一些 LaTeX 的中文化技巧

在这一小节中，我们示范 LaTeX 中文化的一些技巧。

4.6.1 设置节标题的汉字型

标准的 LaTeX 采用 cmbx（粗体字）为节标题的英文字貌，而 gbLaTeX 档又将 cmbx 映射至中黑体，所以节标题的汉字将以中黑体出现。你可以利用 *LaTeX Companion* 书中所述的技巧来更改节标题使用的中英字貌 [2, pp 27–31]。

例 3 若想用粗圆体替换中黑体作为节标题的汉字貌，你可以采用以下的设置方式：

```
\newcommand{\seccface}{\PUXfacematch\PUXFbr}  % 节标题的汉字貌
\renewcommand{\section}{\@startsection
  {section}%                                  % the name
  {1}%                                        % the level
  {0mm}%                                      % the indent
  {-1\baselineskip}%                          % the beforeskip
  {0.5\baselineskip}%                         % the afterskip
  {\bfseries\Large\seccface}}%                % the style
\renewcommand{\subsection}{\@startsection
  {subsection}%                               % the name
  {2}%                                        % the level
  {0mm}%                                      % the indent
  {-\baselineskip}%                           % the beforeskip
  {0.5\baselineskip}%                         % the afterskip
  {\bfseries\large\seccface}}%                % the style
\renewcommand{\subsubsection}{\@startsection
  {subsubsection}%                            % the name
  {3}%                                        % the level
  {0mm}%                                      % the indent
  {-\baselineskip}%                           % the beforeskip
  {0.5\baselineskip}%                         % the afterskip
  {\bfseries\normalsize\seccface}}%           % the style
```

4.6.2 中式节序列号

如果想用中式数字作为节序列号，你可以参考 [2, p.24] 所述的技巧。比方说，以下三个宏的重新定义：

```
\renewcommand{\thesection}{\cnumber{section}}
\renewcommand{\thesubsection}{\thesection.\cnumber{subsection}}
\renewcommand{\thesubsubsection}{\thesubsection.\cnumber{subsubsection}}
```

会使得

```
\section{ 简介 }
\subsection{ 零与一 }
\subsubsection{ 简史 }
```

产生类似如下的结果：

一　简介
一.一　零与一
一.一.一　简史

4.6.3 设置中式的页首标题

你可以利用以下的 LaTeX 命令来添加页首标题：

```
\pagestyle{headings}
```

或利用 fancyhdr 套件（原名为 fancyheading.sty[2, p. 224]）来设置中式的页首标题。譬如底下是一个适用于 gbkreport 与 gbkbook 的设置方式：

```
\usepackage{fancyhdr}
\pagestyle{fancyplain}
\renewcommand{\chaptermark}[1]{%
   \markboth{\twchaptername\ \ \ #1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ \ #1}}
\lhead[\fancyplain{}{\bfseries\thepage}]%
    {\fancyplain{}{\bfseries\rightmark}}
\rhead[\fancyplain{}{\bfseries\leftmark}]%
    {\fancyplain{}{\bfseries\thepage}}
\cfoot{}
```

以上的设置将会产生类似下面所示的页首标题：

11.1 基本概念 （奇数页） **391**

392 （偶数页） 第十一章 动画制作

4.6.4 中文的定理标题

你可以仿照下面的定义方式，把定理类（Theorem-like, [3, p. 79]）环境的标题中文化：

```
\newtheorem{thm}{定理}
```

有此定义之后，下列四行的输入：

```
\begin{thm}
假定 $a$, $b$, $c$, 和 $n$ 均为正整数。当 $n\geq 3$ 时，$a^n + b^n \neq c^n$
成立。此为 {\PUXFmb 费玛最后定理 }。
\end{thm}
```

可得到如下的结果：

定理 **1** 假定 a, b, c, 和 n 均为正整数。当 $n \geq 3$ 时，$a^n + b^n \neq c^n$ 成立。此为费玛最后定理。

你可以用汉字貌匹配命令（参见 5.2.2 节）来改变定理类环境所使用的汉字貌。譬如：若再定义一个环境，将 thm 内的中文改用 `\PUXFmfs`（中仿宋）：

```
\newenvironment{nthm}{\begin{thm}\PUXfacematch\PUXFmfs}{\end{thm}}
```

则

```
\begin{nthm}
假定 $a$, $b$, $c$, 和 $n$ 均为正整数。当 $n\geq 3$ 时，$a^n + b^n \neq c^n$
成立。此为 {\bf 费玛最后定理 }。
\end{nthm}
```

将可得到如下的结果：

定理 **2** 假定 a, b, c, 和 n 均为正整数。当 $n \geq 3$ 时，$a^n + b^n \neq c^n$ 成立。此为费玛最后定理。

其中大部分的中文均改成了〝中仿宋体〞。

4.6.5 其他

由于汉字笔画远比英文字母来得复杂，因此 LaTeX 默认的单行间距往往使得中文档的页面过于拥挤。你可以重新定义 `\baselinestretch` 来增加行距以避免这个问题。笔者建议以下的设置：

```
\renewcommand{\baselinestretch}{1.2}
```

或

```
\renewcommand{\baselinestretch}{1.25}
```

最后，本手册的范例均以反白的例 为标志，并以节为范围来编号。此范例环境的设置方式如下所示：

```
\PUXcfacedef\PUXeg=eg 中楷 s=v
\newfont{\egfont}{CFONTeg11}
\newfont{\egcntfont}{CFONTtb10}
\newcounter{example}[section] \setcounter{example}{1}
\newenvironment{example}{\bigskip\noindent%
  \refstepcounter{example}%
  {\egfont 例 }\ \ {\bf\small \thesection--\theexample}%
  \quad\small\egcntfont}{\medskip}
```

5 pUpTeX 基本命令与内部参数

针对中文排版的特性，pUpTeX 增加了一些基本命令和内部参数。在这一节中，我们介绍提供给常规用户的命令与参数，下一节介绍提供给 TeX 格式档（format）或 LaTeX 套件制作者的命令与参数。

为了避免与现有 TeX/LaTeX 命令的名称起冲突，我们采用底下的字头规则来命名 pUpTeX 命令与参数：

- 命令的名称均以大写英文 `PUX`（代表 pUpTeX eXtension）起头。
- 区域性（local）参数的名称以小写英文 `pux` 起头。这些参数的作用范围仅及于设置的群组（group）之内。
- 全域性（global）参数的名称以小写英文 `puxg` 起头。这些参数的作用范围涵盖整个文件范围。

在表格 2 和 3 中，我们按照字母序列出这些参数与命令的名称、用途、型态、以及本手册中的引用章节。其中型态栏中的字母含意如下：

G 命令或参数具有全域性的影响力。其效力将维持至重新定义为止。

L 命令或参数具有区域性的影响力。其效力不会影响外部群组。

D 命令或参数的影响力与范畴无关（don't care）。

O 参数仅能更改其值一次（once）。

除此之外，pUpTeX 也扩充了 TeX 的 `\font` 命令的语法与功能，使其能够以类似定义英文字体的方式来定义汉字型。这样做的目的是为了让 pUpTeX 也能够使用 LaTeX 2_ε 的 NFSS2 规格。明细请参见第 5.2.4 节

5.1 关闭汉字元的读入

传统 TeX 使用码值 0 – 127 的 ASCII 字符。汉字元由两个 bytes 所组成，而且第一个 byte 的值一定大于 128。pUpTeX 即根据这两个规则来分辨出输入档中单词节的英文字符与双字节的汉字元。然而，许多欧洲国家的字符集通常包含 256 个字符，使用到 129 – 255 的码值，因此文件中若包含这些国家的字符时，就会造成 pUpTeX 解读汉字元的困难。又有少部分 LaTeX 套件包括码值超过 128 的字符，也会造成 pUpTeX 处理上的错误，如 `url.sty` 与使用到它的 `hyperref.sty`。为了克服这类问题，pUpTeX 提供以下控制汉字元读取方式的参数：

`\puxCJKinput`[`=`]0 | 1

名称	用途	型态	参阅
\puxCharSet	设置文件的字符集	L	6.1 节
\puxCJKinput	控制是否读入 CJK 字符	L	5.1 节
\puxgCEspace	调整所有的中英字间距	G	5.4.4 节
\puxgCfaceDepth	调整所有的汉字貌深度	G	5.3.1 节
\puxgCspace	调整所有的汉字间距	G	5.4.1 节
\puxgRotateCtext	逆时钟旋转汉字 90 度	O	5.9 节
\puxCJKcharOther	设置汉字元的类性	L	5.7 节
\puxXspace	保留空白字符	L	7 节

表格 2: PUTEX 新建的参数（依字母序排列）

（方括号内的项目表示可省略，0|1 表示 0 或 1 两者择一。）若设为 1（默认），则正常解读汉字元；若设为 0，则取消解读汉字元，而将其视为两个 bytes。

例 –1 如果你查看本文的 LATEX 原始档，可以发现以下几行：

```
\puxCJKinput=0    % 关闭中文输入，避免读入 url.sty 时发生错误
\usepackage[dvipdfm,
  pdftitle={PUTeX User's Guide},
  pdfauthor={Chey-Woei Tsay},
  pdfpagemode=UseOutlines,
  bookmarks,bookmarksopen,
  pdfstartview=FitH,
  colorlinks, linkcolor=blue,citecolor=blue,
  urlcolor=red,
  ]
  {hyperref}
\puxCJKinput=1    % 重新打开中文输入
```

例 –2 当中文输入处于关闭的状态时，你可以用已经定义成产生汉字句的宏或 \PUXchar 命令（后面会介绍）来输入少量的中文，如下面所示的方式：

```
\newcommand{\manual}{手册}
\puxCJKinput=0
PUTeX \PUXchar"CAB9\PUXchar"D3C3\manual
\puxCJKinput=1
```

会产生如下的结果

PUTeX 使用手册

注：此处的“使”字必须用“\PUXchar"CAB9”而不能用“\PUXchar‘\使”来产生。

名称	用途	型态	参阅
\PUXacnumber	转换成中文数字	D	5.5 节
\PUXcadcode	设置汉字元的类别码	L	5.7 节
\PUXcespace	插入中英文空白间距	D	5.4.7 节
\PUXcfacecespace	调整字貌的中英字间距	G	5.4.5 节
\PUXcfacecspace	调整字貌的汉字间距	G	5.4.2 节
\PUXcfacedef	定义汉字貌	G	5.2.1 节
\PUXcfacedepth	调整汉字貌深度	G	5.3.2 节
\PUXcfontcespace	调整字体的中英字间距	G	5.4.6 节
\PUXcfontcspace	调整字体的汉字间距	G	5.4.3 节
\PUXchar	输入汉字元内码	D	5.6 节
\PUXchardef	定义汉字元命令	L	5.6 节
\PUXcjknumber	转换成中文数字	D	5.5 节
\PUXcnumber	转换成中文数字	D	5.5 节
\PUXcspace	插入中文空白间距	D	5.4.7 节
\PUXdumpfontinfo	输出字体信息	D	7 节
\PUXexspace	插入英文空白间距	D	5.4.7 节
\PUXfacematch	匹配中英文字貌	L	5.2.2 节
\PUXfcnumber	转换成中文数字	D	5.5 节
\PUXfontmatch	匹配中英文字体	L	5.2.5 节
\PUXlocalnames	定义特殊字符表	L	5.7 节
\PUXrangecadcode	设置汉字元区间的类别码	L	5.7 节
\PUXrangetypecode	设置汉字元区间的型态码	L	6.2 节
\PUXspace	插入英文空白间距	D	5.4.7 节
\PUXscnumber	转换成中文数字	D	5.5 节
\PUXsetdefaultcface	设置默认的汉字貌	L	5.2.2 节
\PUXtypecode	设置汉字元的型态码	L	6.2 节
\PUXucnumber	转换成中文数字	D	5.5 节

表格 3: pUTEX 新建的命令（依字母序排列）

最后，我们要特别提醒读者：当你关闭中文输入后，在同一群组使用中文之前记得要重新打开，否则就无法正确读入汉字元了。

5.2 汉字貌与字体

与其他的中文 TEX 系统相比，PUTEX 提供最强大却最容易使用的汉字型处理技术。你可以使用任何中文 TrueType 字体，而不再受限于少少几套的汉字型。你也不需要在硬盘中安装颇占空间的中文 bitmap 字体。此外，你只需使用几行命令就能在 TEX/LATEX 中轻松地定义与使用各式各样的汉字型。

在说明如何设置汉字型之前，我们先区别字貌（font face）与字体（font）这两个术语。简单地说：所谓字貌即字的外观，而字体等同于字貌再加上大小宣告。PUTEX 的汉字貌是由底下的属性来定义：

字体（**font look**） 区别出字的形状。以中文为例，宋体、楷书体、和中黑体是三种不同的字体。

字重（**font weight**） 用以设置字的粗细。中文 TrueType 字体可以定义出 9 种粗细程度。

字样（**font shape**） 用以设置字的样式，如正体（normal）、斜体（italic）、反白体（reversed）、与旋转体（rotated）。

5.2.1 定义汉字貌

PUTEX 的 `\PUXcfacedef` 命令用来定义汉字貌。它的语法如下：

`\PUXcfacedef`*faceid* [=] *ename cname* [*attributes*]

（方括号内的项目表示可以省略），其中各参数的意义如下：

faceid 代表此汉字貌的命令。你可以用此命令来改变当前使用的汉字貌（参见 5.2.3 节）。

ename 指定汉字貌的英文代名（identifier）。此名称必须全由（大小写）英文字母组成，不得使用其他的字符。也不得重复使用。

cname 指定汉字体的逻辑名称（如细明、中楷等）。此名称可用中文来命名。你必须在 cfonts.map 档中定义其所映射的真实汉字体（参见附录 A），否则此字貌将使用默认的真实中文字体（通常为宋体），而无法打印出你所期望的字体。

attributes 这一个参数选项列指定以下的字貌属性（可以任意组合和以任意的顺序宣告）：

100	特细	200	中细	300	细
400	正常	500	稍粗	600	中粗
700	粗	800	特粗	900	超粗

表格 4: 字重属性值映射的粗细程度

`t=100|···|900` 此属性设置字貌的粗细（字重）。属性值所映射的粗细程度如表格 4 所示。若没有宣告此属性的话，则字貌默认为正常的粗细（即 $t = 400$）。

`d=`n 设置字貌深度（depth）为字体大小的 $n/1000$。

`s=i` 设置字貌为斜体字。

`s=r` 设置逆时针旋转 90 度的字貌。

`s=v` 设置为反白字貌。

底下是一些定义汉字貌的例子：

```
\PUXcfacedef\PUXFbody=body bodyface d=170   % bodyface 汉字貌，字深 0.17
\PUXcfacedef\PUXFtmb=tmb 细明 t=700         % 细明粗体汉字貌
\PUXcfacedef\PUXFtmi=tmi 细明 s=i           % 细明斜体汉字貌
\PUXcfacedef\PUXFtmv=tmv 细明 s=v           % 细明反白体汉字貌
\PUXcfacedef\PUXFtmr=tmr 细明 s=r           % 细明旋转体汉字貌
\PUXcfacedef\PUXFmrrv=mrrv 中圆 s=r s=v     % 中圆旋转反白体汉字貌
```

为了方便起见，gbTEX/LATEX 内建了若干常见的汉字貌。这些字貌均采用默认的属性值。表格 5 列出这些字貌的逻辑名称与英文名称。

细明	tm	中明	mm	粗明	bm	特明	sm
细黑	tb	中黑	mb	粗黑	bb	特黑	sb
细圆	tr	中圆	mr	粗圆	br	特圆	sr
细楷	tk	中楷	mk	粗楷	bk	特楷	sk
细仿宋	tfs	中仿宋	mfs	粗仿宋	bfs	特仿宋	sfs
细隶书	tli	中隶书	mli	粗隶书	bli	特隶书	sli
细行书	tsn	中行书	msn	粗行书	bsn	特行书	ssn
细叠圆	tdr	中叠圆	mdr	粗叠圆	bdr	特叠圆	sdr
勘亭流	kd	古印	gn	综艺	je	魏碑	wb
POP1	po	颜体	yt	俪中宋	lms	少女	girl
`default`	default						

表格 5: gbTEX/LATEX 内建的汉字貌

为了方便记忆，除了字貌 `default` 以外，笔者用下面两个规则来命名字貌的英文名称：

1. 如果字貌集有粗细之别（如细明、中明、粗明、和特明），则第一个字母用 t、m、b、和 s 四个字母分别代表细（thin）、中（medium）、粗（bold）、和特（super）四种粗细。如果无粗细之别，则不加此指示粗细的字母。
2. 在代表粗细的字母之后，使用一个或两个英文字母来代表汉字体，如 m 代表明体、b 代表黑体、k 代表楷书、r 代表圆体、 fs 代表仿宋体等等。

此外，若字貌的英文名称为 xx，则命令命名为 `\PUXF`xx （`PUXF` 中的字母 `F` 代表 Face 之意）。比方说：若英文名称是 mm 的字貌，其字貌命令则命名为 `\PUXFmm`。

当你添加自定的汉字貌，命名时，请务必避开这些内建字貌名称，否则会造成重复定义的错误。譬如：

`\PUXcfacedef\a=tm` 细明	错误！与内建的字貌 tm 重名
`\PUXcfacedef\b=mk` 中楷	错误！与内建的字貌 mk 重名
`\PUXcfacedef\PUXFmk=mk` 中楷	错误！与内建的字貌 mk 重名

5.2.2 定义中英字貌的匹配规则

中/英字貌的匹配规则可说是 PUfTEX 的核心观念之一，因为透过这项技巧，汉字型变得更容易设置与使用。比方说： LATEX 英文字貌默认的用法为：cmr 用于正常字、cmtt 用于定宽字、cmbx 用于粗体字、以及 cmti 用于斜体字。如果我们设置以下四个匹配规则：

- 英文用 cmr 字貌时，中文则用 `\PUXFtm` 字貌。
- 英文用 cmtt 字貌时，中文则用 `\PUXFtm` 字貌。
- 英文用 cmbx 字貌时，中文则用 `\PUXFmb` 字貌。
- 英文用 cmti 字貌时，中文则用 `\PUXFmk` 字貌。

就可以轻易地获得以下的汉字型变化效果：

- 正常的汉字使用宋体。
- 出现在 `\tt` 环境的中文使用宋体。
- 出现在 `\bf` 环境的中文使用中黑体。
- 出现在 `\em` 和 `\it` 环境的中文使用中楷体。
- 改变英文字体大小的命令（如 `\small` 和 `\Large`）也同时改变汉字型的大小。

上述中/英字貌的匹配规则必须利用 PUfTEX 的 `\PUXfacematch` 命令来达成。此命令的第一种格式如下：

\PUXfacematch *eface_name* *cface_id*

其中，参数 *eface_name* 是英文字貌的名称，*cface_id* 是汉字貌命令。命令的作用将使得出现在英文字貌 *eface_name* 环境中的汉字元，均采用 *cface_id* 所代表的汉字貌，且其字体大小与英文字体相同。比方说，底下的命令将英文字貌 cmr 匹配成 \PUXFtm（细明）：

```
\PUXfacematch cmr \PUXFtm
```

用白话一点的句子来说就是：“当英文使用 cmr 字体时，中文就使用宋体。”

为了方便制作中文 TEX/LATEX 文件，gbTEX/LATEX 预先定义了下列中/英字貌匹配规则：

```
\PUXfacematch cmr  \PUXFtm    % 正常字为宋体
\PUXfacematch cmtt \PUXFtm    % 定宽字为宋体
\PUXfacematch cmbx \PUXFmb    % 粗体字为中黑体
\PUXfacematch cmti \PUXFmk    % 斜体字为中楷体
\PUXfacematch cmbxti \PUXFmk  % 粗斜体用中楷体
```

这些规则让用户可以用字体变化命令（如 \rm, \em, \it, \bf, 等）来选择汉字型；也可以用字体大小变化命令（如 \small, \large, \huge 等）来改变中文的大小。举例来说，以上的字貌匹配可以得到下表所示的 LATEX 文字变化效果：

输入	打印结果
正常文字 normal	正常文字 normal
{\bf bold（粗体字）}	**bold**（粗体字）
{\it italic（斜体字）}	*italic*（斜体字）
{\em emphasize（强调字）}	*emphasize*（强调字）
{\small 小字}	小字
{\Large 大字}	大字
{\Huge 巨字}	巨字

例 –3 如果你想把正常汉字改用细圆体，只需将 cmr 的匹配汉字貌改成 \PUXFtr 即可。比方说，你可以在 LATEX 文件中键入：

```
\documentclass...
\PUXfacematch cmr \PUXFtr     % 正常汉字改用细圆体
```

或在 TEX 文件的第一行键入：

```
\PUXfacematch cmr \PUXFtr     % 正常汉字为细圆体
```

例 –4 你可以自行定义其他的中英字貌匹配。假定你想将 cmbxti（粗斜体英文字貌）与粗黑体汉字貌匹配，可以在 LaTeX 文件中键入：

```
\documentclass...
\PUXfacematch cmbxti \PUXFbb
```

或在 TeX 文件的第一行键入：

```
\PUXfacematch cmbxti \PUXFbb
```

若汉字元出现在未设置汉字型也未定义匹配的英文字貌环境中，gbTeX/LaTeX 会自动选择 `default` 字貌，换句话说， `default` 就是所谓的默认汉字貌。你可以用命令：

\PUXsetdefaultcface*cface_id*

改选 *cface_id* 为默认的汉字貌。譬如：

```
\PUXsetdefaultcface\PUXFmk
```

使得字貌 `\PUXFmk`（中楷）替换 `default` 成为默认的汉字貌。

匹配英文 **PostScript** 字貌

底下，我们示范如何匹配英文 PostScript 字貌，也介绍 `\PUXfacematch` 的第二种格式。此处我们假定你熟悉 LaTeX 的 NFSS 与 PSNFSS，若非如此的话，请先参考 The LaTeX Companion 书中的说明 [2]。

如果想改用 PostScript Times-Roman 来替换 LaTeX 默认的 Computer Modern 英文字貌，你可以采用 `times.sty` 套件（置于`\texmf\tex\latex\psnfss` 目录）。其主要的内容如下：

```
\renewcommand{\sfdefault}{phv}
\renewcommand{\rmdefault}{ptm}
\renewcommand{\ttdefault}{pcr}
```

我们得知：正体字、斜体字、与粗体字是靠 ptm 所产生，接下来参考 `ot1ptm.fd` 字体定义档的内容，它也是置于 `\texmf\tex\latex\psnfss` 目录中。由其中的三行定义：

```
\DeclareFontShape{OT1}{ptm}{m}{n}{<-> ptmr7t}{}
\DeclareFontShape{OT1}{ptm}{m}{it}{<-> ptmri7t}{}
\DeclareFontShape{OT1}{ptm}{b}{n}{<-> ptmb7t}{}
```

可知上述三种字体分别采用 ptmr7t、ptmri7t、与 ptmb7t 字貌。添加以下四行就可以使得细明匹配正体英文、中楷匹配斜体英文、以及中黑匹配粗体英文：

```
\usepackage{times}
\PUXfacematch ptmr7t \PUXFtm
\PUXfacematch ptmri7t \PUXFmk
\PUXfacematch ptmb7t \PUXFmb
```

为了简化上述显然有点复杂的步骤，pUTEX 4 新建底下第二种格式的 \PUXfacematch 命令：

\PUXfacematch*cface_id*

此命令把当前英文字体的字貌与汉字貌 *cface_id* 匹配成对，并依据此新匹配规则更改当前的汉字型。

例 –5 我们可以使用第二型的 \PUXfacematch 命令来设置上述 PostScript 字貌的匹配，而不需要去查英文字貌的名称。譬如：在 LATEX 文件的前端，我们可以做以下的匹配设置：

```
\usepackage{times}
\rmfamily\upshape\mdseries\PUXfacematch\PUXFtm % 罗马体 → 细明（常规文字）
\rmfamily\upshape\bfseries\PUXfacematch\PUXFbm % 粗罗马体 → 粗明（标题文字）
\rmfamily\itshape\mdseries\PUXfacematch\PUXFmk % 斜体 → 中楷
\rmfamily\itshape\bfseries\PUXfacematch\PUXFbk % 粗斜体 → 粗楷
\ttfamily\upshape\mdseries\PUXfacematch\PUXFtm % 定宽字 → 细明
\sffamily\upshape\mdseries\PUXfacematch\PUXFmb % 黑体字 → 中黑
\sffamily\upshape\bfseries\PUXfacematch\PUXFbb % 粗黑体字 → 粗黑
% 其他英文字貌映射至默认的汉字貌
```

（上面 LATEX 字体选择命令的意义，请参考 [3, p.66]。）

退出这一小节之前，我们用下面的例子来说明 \PUXfacematch 命令的区域性效果：

例 –6 以下的输入：

```
{\rm 细明☆ {\rm 还是细明☆ \PUXfacematch\PUXFmk 变成中楷 } ☆又回到细明 }
```

产生的结果为：

细明☆还是细明☆变成中楷☆又回到细明

因为这种区域性效果，所以上一个例中，

```
\rmfamily\upshape\mdseries\PUXfacematch\PUXFtm % 罗马体用细明（常规文字）
```

不能用底下的方式替换：

```
\textrm{\PUXfacematch\PUXFtm} % 匹配只在 {...} 中有效，而没有全域性的效果
{\rmfamily\upshape\mdseries\PUXfacematch\PUXFtm} % 理由同上
```

5.2.3 改变当前使用的汉字貌

在 \PUXcfacedef 命令中所定义的汉字貌命令（如 \PUXFmk）可以用来改变当前群组内的汉字貌（汉字型的大小则依当前的英文字体而定）。字貌命令是一种局部性命令，它的效力涵盖所处的群组与其内部群组，但不包含外部群组。在其效力范围内，汉字貌命令会取消中/英字貌的匹配设置。

例 –7 以下的输入方式：

```
宋体、{\PUXFmk 中楷体 }、{\PUXFmb 中黑体 }
```

产生的结果为：

宋体、中楷体、中黑体

例 –8 底下的两种输入方式：

```
正常、{\small\PUXFmk 中楷体 }、{\huge\PUXFmk 中楷体 }
正常、{\PUXFmk\small 中楷体 }、{\PUXFmk\huge 中楷体 }
```

会产生相同的结果：

正常、中楷体、中楷体

例 –9 底下的输入方式：

```
{\PUXFtm 正常、{\em 强调字 }、{\bf 粗体字 }}
```

产生的结果为：

正常、强调字、粗体字

（都是宋体）而不是：

正常、强调字、粗体字

因为字貌命令 \PUXFtm 使得群组内的字貌匹配失效。如果想获得上一行的效果，你可以采用底下的输入方式：

```
{\PUXFtm 正常、{\PUXFmk 强调字 }、{\PUXFmb 粗体字 }}
```

5.2.4 定义汉字型

在 TEX 中，我们用 \font 命令来定义一个英文字体。譬如：

```
\font\tenrm cmr10
```

这一行命令定义 \tenrm 为 cmr10 的字体。大部份的 TEX 字体名称是由字貌名称与字体大小所组成。以 cmr10 为例，cmr 为其字貌名称，而 10 表示其字体大小为 10pt。PUTEX 即根据此观察来扩充 \font 命令的功能，使之能用于定义中文的字体。其用法如下：

\font*cfontid* CFONT*fn*

其中，*cfontid* 是用户自定义的汉字型命令名称；CFONT*fn* 则是由三个部份所组成：

CFONT	这个开头英文字告知 PUTEX：此处将定义汉字型，而非 TEX 字体。
f	是一个已定义之汉字貌的英文名称。
n	是一个用来宣示字体大小的正整数。

底下是定义汉字型的一些范例（此处采用表格 5 的汉字貌）：

```
\font\a CFONTtm10               % \a 是 10pt 的宋体
\font\b CFONTtm24               % \b 是 24pt 的宋体
\font\c CFONTmk18               % \c 是 18pt 的中楷体
\font\d CFONTmfs12 at 14pt      % \d 是 14pt 的中仿宋体
\font\e CFONTsb12 scaled 2000   % \e 是 24pt 的特黑体
```

如果用 LATEX 的 \newfont 命令来定义字体的话，以上的例子可改写成：

```
\newfont{\a}{CFONTtm10}               % \a 是 10pt 的宋体
\newfont{\b}{CFONTtm24}               % \b 是 24pt 的宋体
\newfont{\c}{CFONTmk18}               % \c 是 18pt 的中楷体
\newfont{\d}{CFONTmfs12 at 14pt}      % \d 是 14pt 的中仿宋体
\newfont{\e}{CFONTsb12 scaled 2000}   % \e 是 24pt 的特黑体
```

5.2.5 设置中英文字体的匹配

PUTEX 的字貌匹配技术虽然简化了汉字型的设置与使用，但是完全不考虑字体大小的因素，有时也会带来一些困扰。PUTEX 因而提供 \PUXfontmatch 命令让用户可以做更精确细微的字体匹配。\PUXfontmatch 命令也有两种形式，其语法分别如下：

1. \PUXfontmatch*cfaceid*

2. `\PUXfontmatch` *efont* *cfont*

第一种形式是用来匹配当前使用之英文字体。匹配的汉字型将以 *cfaceid* 为其字貌，并以英文字体的大小为其大小。第二种形式中的 *efont* 是一个英文字体的名称、 *cfont* 是一个汉字型的名称。它的作用是让出现于 *efont* 英文字体环境内的中文采用 *cfont* 汉字型。

例 –10 由于 `\PUXfontmatch` 是一个区域性命令，因此以下的输入：

```
{\em 强调☆ {\em\PUXfontmatch\PUXFtm 强调 } ☆强调 }
```

所得到的结果为：

强调☆强调☆强调

而非

强调☆强调☆强调

例 –11 假定你做如下的字体匹配设置：

```
\font\A cmr10
\font\B CFONTmk10
\PUXfontmatch \A \B
```

这会使得

```
{\A cmr10 font 和 中楷体 10pt 字体 }
```

产生底下的排版结果：

cmr10 font 和中楷体 10pt 字体

其中的中英文字皆为 10pt 大小。

例 –12 你可以匹配大小不同的中英字体。譬如：

```
\font\A cmr10
\font\B CFONTmk20
\PUXfontmatch \A \B
```

将使得

```
{\A cmr10 font 和 中楷体 20pt 字体 }
```

产生以下的结果：

cmr10 font 和中楷体 20pt 字体

其中的英文字为 10pt 大小，而汉字为 20pt 大小。

5.3 调整汉字貌深度

基线（baseline）是一条用来排列文字的虚拟水平线。字体深度（depth）决定基线穿越文字的位置。由于 TrueType 汉字型的深度通常为字体大小的 0.2，而 TEX 字体的深度则为字体大小的 0.25，两者的差距有时会造成中英文并置时，高矮不协调。因此 PUTEX 提供参数 \puxgCfaceDepth 让你调整所有汉字的深度；提供命令 \PUXcfacedepth 让你调整特定汉字貌的文字深度。调整深度时，你应该先设置 \puxgCfaceDepth的值，然后才用 \PUXcfacedepth 设置个别汉字貌的文字深度。

由于 PUTEX 将深度视为字貌的属性，因此你在文件前端设置好字貌深度后，请勿在文件中加以改变，否则会产生无效的排版结果。

5.3.1 全域性地调整汉字深度

PUTEX 的参数 \puxgCfaceDepth 是用来调整中英文并置对齐的高度。其格式如下：

\puxgCfaceDepth[=]n

整数值 n 用来设置深度。n 值愈大，则汉字位置就愈低，反之则愈高。假定字体大小为 s，深度的计算公式如下：

$$\text{depth} = s \times n/1000$$

\puxgCfaceDepth 参数的默认是 200（即采用 TrueType 的深度值 0.2）。此设置尚能获得良好的排版结果。如果你不满意此默认，你可以改变它。常规而言，范围应在 150 至 200 之间。

5.3.2 调整汉字貌深度

PUTEX 的 \PUXcfacedepth 命令可用来调整个别汉字貌的文字深度。其语法如下：

\PUXcfacedepth*cfaceid*=n

其中的 *cfaceid* 是一个汉字貌命令。 \PUXcfacedepth 是一个全域性的命令。设置汉字貌 *cfaceid* 的深度后，其效果将维持至重新设置 *cfaceid* 的深度为止。

例 **–13** 华康隶书体中文与 TEX cmr 字体并置时，会显得过高。你可以用以下的命令：

```
\PUXcfacedepth\PUXFmli=250
```

将隶书体的深度调整成 0.25，使其位置往下移。

5.4 调整文字的间距

pUfTEX 提供一些参数让你调整中文的字间距以及中文与英文的间距。调整的对象可以是所有的汉字貌、单一的汉字貌、或单一的汉字型。然而，你应该按照

一、 调整所有文字的间距

二、 调整单一汉字貌的文字间距

三、 调整单一汉字型的文字间距

这个顺序下达命令，否则会造成调整无效。比方说：若你先调整某一字体的文字间距，然后再调整该字体所属字貌的文字间距，则后者的设置将替换前者的设置，而使得前者的设置无效。

所有调整文字间距的命令都是全域性，设置之后，效果将维持至重新更改其值为止。为了保持文字间距的协调与统一，你最好只在文件前端设置间距来调整文件整体的紧密程度。若要局部调整，你应该使用 `\hspace`、`\␣`、或 `$\,$` 之类的 LaTeX 命令。

5.4.1 调整所有汉字的间距

参数 `\puxgCspace` 是用来调整所有汉字的间距。它的语法如下：

`\puxgCspace`[=]w [`plus` m `minus` n]

其中的 w, m, 和 n 是三个整数值，分别称为空白间距的自然宽度（natural width）、加量（stretch）、与减量（shrink）。w 值愈大，间距就愈宽，反之则愈窄；m 是空白间距可延伸的建议最大值（因为可能会超过此值，所以不称为最大值）；n 是空白间距可缩短的最大值。想进一步探究这些术语与概念的读者，请参考 [4, p.69]。

假定 s 是当前汉字型的大小，则该字体的空白宽度计算公式如下：

$$\begin{aligned} \text{width} &= s \times w/1000 \\ \text{stretch} &= s \times m/1000 \\ \text{shrink} &= s \times n/1000 \end{aligned}$$

为了与旧版的 pUfTEX 兼容，若不设置加量与减量，计算公式则变成：

$$\begin{aligned} \text{width} &= s \times w/1000 \\ \text{shrink} &= s \times |w|/3000 \\ \text{stretch} &= \begin{cases} 250s/2000 & \text{if } w < 250 \\ s \times w/2000 & \text{if } w \geq 250 \end{cases} \end{aligned}$$

`\puxgCspace` 的默认是 50（不设置加量与减量）。

例 –**14** 若你想把空白宽度设成约为字体宽的 1/5，可以宣告 `\puxgCspace=200`（因为 200/1000 = 1/5）。若你希望仅以汉字型本身的周围空白来分隔文字，可以宣告 `\puxgCspace=0`。

例 –**15** 假定 s 是汉字型的大小。底下设置的效果如注解中的说明：

```
\puxgCspace=50 plus 0 minus 10   % 空白宽度设在 0.04s 到 0.05s 之间
\puxgCspace=50 plus 10 minus 0   % 空白宽度设为最少 0.05s
\puxgCspace=50 plus 0 minus 0    % 空白宽度固定为 0.05s
```

我们不建议你设置固定的空白宽度（如上例的第三项），因为这会造成 TEX 程序不容易找到适当的断行点，使得许多文字行不是太长就是太短。以下是笔者建议设置汉字间距的三个步骤：

1. 先把 w 值设置成最适当的汉字间距宽度。
2. 选择 n 的值使得 $w - n$ 是能允许的最小汉字间距。
3. 若想让 TEX 在断行时把多余空白主要分配给中文间距，则把 m 值设大一点，否则，设小一点。

然而在大部分的情况下，使用旧式的设置方式（省略加量与减量）就可以得到不错的效果。

5.4.2 调整汉字貌的文字间距

`\PUXcfacecspace` 命令用来调整个别汉字貌中文间距。其语法如下：

`\PUXcfacecspace\`*cfaceid*[=]w [`plus` m `minus` n]

其中的 `\`*cfaceid* 是一个汉字貌命令，前余的参数 w, m, 和 n 如前一小节所述。

例 –**16** 由于楷书体的字体比较小，使得楷书体的文字间距看起来宽了些，因而你可以用命令：

```
\PUXcfacecspace\PUXFmk=-50
```

将空白宽度设成约为字体大小的 −0.05 倍以缩短文字间距。

5.4.3 调整汉字型的文字间距

`\PUXcfontcspace` 命令用来调整个别汉字型的字间距。它有以下两种型式：

`\PUXcfontcspace\`*cfontid*[=]*w* [`plus` *m* `minus` *n*]
`\PUXcfacecspace\font`[=]*w* [`plus` *m* `minus` *n*]

第一式调整汉字型 `\`*cfontid* 的文字间距；第二式调整当前汉字型的文字间距（此处，`\font` 命令被解读成当前汉字型，而不是 TEX `\font` 命令的原意）。

不同于字貌，字体的大小是固定的，所以设置字体的文字间距时，`\PUXcfontcspace` 命令中的参数 w, m, 和 n 必须加上长度单位。下表列出常用的长度单位：

`pt`	point ($1\,\mathrm{pt} = 1/72.27\,\mathrm{in}$)
`in`	inch ($1\,\mathrm{in} = 72.27\,\mathrm{pt}$)
`cm`	厘米 ($2.54\,\mathrm{cm} = 1\,\mathrm{in}$)
`mm`	公厘 ($10\,\mathrm{mm} = 1\,\mathrm{cm}$)
`ex`	当前英文字体字母 x 的高度（约等于字体高度的一半）
`em`	当前英文字体字母 M 的宽度（约等于字体宽度）

（更完整的表列请参见 [4, p.56]。）

例 17 以下的设置：

```
\font\A CFONTmk12
\PUXcfontcspace\A=0.72pt plus 20pt minus 6pt
```

将 12pt 中楷体字体 `\A` 的汉字间距的自然宽度设为 0.72 pt、加量为 20 pt、和减量为 6 pt。

5.4.4 调整中文与英文的字间距

PUTEX 的参数 `\puxgCEspace` 是用来调整所有汉字与英文字的间距。它的语法如下：

`\puxgCEspace`[=]*w* [`plus` *m* `minus` *n*]

请参考 5.4.1 节关于 w, m, 和 n 的说明。

假定 s 是当前汉字型的大小，则该字体的中英空白宽度计算公式如下：

$$\begin{aligned} \text{width} &= s \times w/1000 \\ \text{stretch} &= s \times m/1000 \\ \text{shrink} &= s \times n/1000 \end{aligned}$$

为了与旧版的 pUfTEX 兼容，若不设置加量与减量，计算公式则变成：

$$\begin{aligned} \text{space} &= s \times w/1000 \\ \text{shrink} &= s \times |w|/3000 \\ \text{stretch} &= s \times |w|/2000 \end{aligned}$$

\puxgCEspace 的默认是 150（不设置加量与减量）。

例 –18 若你想把空白宽度变为汉字型宽度的 1/5，可以宣告 \puxgCespace=200。若你希望仅以汉字型本身的周围空白来分隔中英文字，可以宣告 \puxgCespace=0。

5.4.5 调整汉字貌的中英字间距

命令 \PUXcfacecespace 可用来调整个别汉字貌的中英字间距。其语法如下：

\PUXcfacecespace*cfaceid*[=]*n* [plus *m* minus *n*]

其中的 *cfaceid* 是一个汉字貌命令。

5.4.6 调整汉字型的中英字间距

命令 \PUXcfontcespace 可用来调整个别汉字型的中英字间距。它有以下两种型式：

\PUXcfontcespace*cfontid*[=]*w* [plus *m* minus *n*]
\PUXcfontcespace\font[=]*w* [plus *m* minus *n*]

第一式调整汉字型 *cfontid* 的中英字间距；第二式调整当前汉字型的中英字间距（此处，\font 命令被解读成当前汉字型，而不是 TEX \font 命令的原意）。情参考 5.4.3 节中关于参数 w, m, 和 n 的说明。

例 –19 以下的设置：

```
\font\A CFONTmk12
\PUXcfontcespace\A=0.72pt plus 20pt minus 6pt
```

将 12 pt 中楷体字体 \A 的中英字间距的自然宽度设为 0.72 pt、加量为 20 pt、和减量为 6 pt。

5.4.7 插入空白

若 pUfTEX 无法产生正确的空白，或你想自行插入某种空白间距，这时可以用以下的命令：

\PUXspace 英文空白（如同英文字之间的空白 ␣）

\PUXexspace	英文空白（如同英文字之间的空白命令 \␣）
\PUXcspace	中文空白
\PUXcespace	中英空白
\PUXchar"A1A1	中文空白字符（参见 5.6 节）

5.5 中式数字

pUTEX 提供一些命令把整数值转换成中文数字符串。譬如：\PUXcnumber 123 把 123 转换成字符串〝一百二十三〞。表格 6 列出这些命令的名称。其中，俗体与小写的差异在于：俗体使用〝廿〞和〝卅〞来替换〝二十〞和〝三十〞。正式与大写的差异在于：正式不省略十进制数字（如表格 6 中数值 12 所示）。阿拉伯则是采用阿拉伯数字的 GBK 全角字。\PUXcjknumber 则是把每个阿拉伯数字一一地换成中文数字。它的完整形式如下：

\PUXcjknumber n [offset[=]l]

若省略 offset 部分，则 l 的默认是 0。这个命令假定特殊字符表自第 l 开始存放了 0, 1, ..., 9 阿拉伯数字的本地映射字符（请参阅 6.3 节关于特殊字符表的说明）。以中文为例，0 的中文映射字符是〝〇〞、1 映射至〝一〞、... ... 、等等。然后，\PUXcjknumber 把每一位数转换工本地的映射字符。

命令	格式	12	123	1230531
\PUXcnumber	小写	十二	一百二十三	一百二十三万〇五百三十一
\PUXscnumber	俗体	十二	一百廿三	一百廿三万〇五百卅一
\PUXucnumber	大写	拾贰	壹佰贰拾参	壹佰贰拾参万零伍佰参拾壹
\PUXfcnumber	正式	壹拾贰	壹佰贰拾参	壹佰贰拾参万零伍佰参拾壹
\PUXacnumber	阿拉伯	１２	１２３	１２３０５３１
\PUXcjknumber	中文	一二	一二三	一二三〇五三一

表格 6: 中式数字命令、格式、与范例

例 –20 以下显示各式中文数字的产生方法：

输入	结果
\PUXcnumber 123	一百二十三
\PUXscnumber 123	一百廿三
\PUXucnumber 123	壹佰贰拾参
\PUXfcnumber 123	壹佰贰拾参
\PUXacnumber 123	１２３

pUTEX 4 另新建一个转换数字的命令，可用来把数字 1, 2, 3, ⋯ 转成〝甲、乙、丙、……〞或〝子、丑、寅、……〞。此命令的格式如下：

`\PUXnameseq` n `min`[`=`]a `max`[`=`]b `offset`[`=`]l

其作用是打印出特殊字符表中第 $l+n-a$ 的字符，其中，$n, a, b,$ 和 l 的值必须介于 0 到 255 之间，而且 $a \le n \le b$，否则会产生错误的结果。

例 21 由于特殊字符表自位 54 开始存放甲、乙、丙、…… 等 10 个代表天干的字符，因此：

`\PUXnameseq 1 min=1 max=10 offset=54` 得到字符〝甲〞
`\PUXnameseq 2 min=1 max=10 offset=54` 得到字符〝乙〞
⋮
`\PUXnameseq 10 min=1 max=10 offset=54` 得到字符〝癸〞

例 22 由于特殊字符表自位 64 开始存放子、丑、寅、…… 等 12 个代表地支的字符，因此：

`\PUXnameseq 1 min=1 max=12 offset=64` 得到字符〝子〞
`\PUXnameseq 2 min=1 max=12 offset=64` 得到字符〝丑〞
⋮
`\PUXnameseq 12 min=1 max=12 offset=64` 得到字符〝亥〞

5.6 字符码命令

pUTEX 仿照 TEX 基本命令 `\char` 的功能 [4, p.43]，提供 `\PUXchar` 命令让用户能够用字符码的方式输入汉字元。你可以选用以下的方式输入字符码：

`\PUXchar"`*hhhh* % *hhhh* 是十六进制的字符码
`\PUXchar'`*oooooo* % *oooooo* 是八进制的字符码
`\PUXchar‘\`*c* % *c* 是字符

不论那一种方式，输入的字符码都必须大于 255，否则会产生错误信息。此外，由 `\PUXchar` 命令输入的字符，其类别码都是 12 (other)（参见 5.7 节）。

例 23 `\PUXchar` 命令可用来输入不易从键盘键入的字符，譬如：

`\PUXchar"A1E1` 产生 ⇕
`\PUXchar‘\ㄅ` 产生 ㄅ

注：有些汉字体（如宋体和楷书体）并未包含所有的 GBK 符号字符。

例 –24 中文空白字符的十六进制 GBK 码为 A1A1，所以你可以用 \PUXchar"A1A1 的方式插入一个中文空白字符。譬如输入：姓\PUXchar"A1A1 名，所得的结果为“姓　名”。

pUpTeX 也仿照 TeX 基本命令 \chardef 的功能 [4, p. 44]，提供 \PUXchardef 命令。其格式如下：

\PUXchardef*cmd*="hhhh

其作用是将命令 *cmd* 定义成字符码为 hhhh 的汉字元。

例 –25 底下的命令定义：

\PUXchardef\空="A140

让你可以使用命令“\空”来插入一个中文空白字符“　”。

5.7 设置汉字元的类别码

TeX 读入字符时，会根据字符的类别码（catcode）[4, p.37] 来运行预定的动作。通常英文字母的类别码是 11（代表 letter）、阿拉伯数字和常规标点符号的类别码是 12（代表 other）、其他特殊的符号，如 \ { } $ ··· 等等则被分配特定的类别码。letter 和 other 类的字符通常不引发特殊处理，而直接进入排版阶段。两者不同处在于：letter 字符可用于命令的命名，而 other 字符则不行。 pUpTeX 把正常汉字元的类别默认成 letter，其他符号字符则默认成 other。

TeX 的基本命令 \catcode 可用来区域性地改变某字符的类别码，藉此改变该字符的处理方式。pUpTeX 提供映射的命令 \PUXcatcode：

\PUXcatcode *n*[=]*c*

把码值为 *n* 的汉字元的类别码（区域性地）设成 *c*。针对设置一连串字符的类别码，pUpTeX 提供以下的命令：

\PUXrangecatcode *m* [to] *n* [=] *c*

把字码从 *m* 到 *n* 的字符的类别码都设成 *c*。

例 –26 中文句点（。）的十六进制码值为 A1A3。你可以用底下前四种方式来设置其类别码为 letter：

```
\PUXcatcode‘\。=11       % 用字符码值（注：‘ 是位于键盘左上角的反引号）
\PUXcatcode"A1A3=11      % 用十六进制码值
\PUXcatcode’120643=11    % 用八进制码值
\PUXcatcode 41379=11     % 用十进制码值
\PUXcatcode 。=11        % 错误！
\PUXcatcode’\。=11       % 错误！不能用单引号，必须用反引号
```

当字符的类别码被设成 13 时，此字符变成命令，所以又称为主动字符（active character）。我们以下面的例子来示范利用主动字符来更改中文标点符号的排版方式。

例 –27 图 1 显示两种中文标点的排版方式：(a) 标点置中并占一个全角字符的宽度；(b) 标点往左移若干宽度，句点和逗点往下移。如果你查看本件的原始档，可以看到 (a) 和 (b) 的文字输入完全相同，惟一差别在于：(b) 增加了以下的设置：

```
\PUXcatcode‘\，=\active
\def，{\kern-0.25em\raise-.75ex\hbox{\PUXchar‘\，}\kern-0.2em}
\PUXcatcode‘\。=\active \def。{\kern-0.25em\raise-.75ex\hbox{\PUXchar‘\。}}
\PUXcatcode‘\；=\active \def；{\kern-0.2em\PUXchar‘\；}
\PUXcatcode‘\：=\active \def：{\kern-0.2em\PUXchar‘\：}
\PUXcatcode‘\“ =\active \def“ {\PUXchar‘\“ \kern-0.25em }
\PUXcatcode‘\” =\active \def” {\kern-0.25em\PUXchar‘\” }
```

其中，我们用 \PUXcatcode 把中文标点，。；“ ” 设成主动字符，然后再定义成命令来改变它们的排版位置。举例来说：

```
\def，{\kern-0.25em\raise-.75ex\hbox{\PUXchar‘\，}\kern-0.2em}
```

使得逗号往左移 0.25em、往下移 0.75ex、然后再让后面跟着的字符往左移 0.2em。由于逗号已经是主动字符，所以在定义中必须改用“ \PUXchar‘\，”，否则会造成自我递回定义的错误。

例 –28 我们可以利用主动字符来转换字符。举例来说，以下的设置：

```
\PUXcatcode‘\（=\active \def“ {『}
\PUXcatcode‘\）=\active \def” {』}
```

会使得输入：（TEX）产生结果：『TEX』，因为（和）分别被『和』给替换掉了。

例 –29 在 verb 与 verbatim 环境中，PUTEX 仍会在中英文字符之间添加中英空白。有时，这会造成字符间距过大，譬如：

```
\begin{verbatim}
    \newcommand{\manual}{手册}
\end{verbatim}
```

我说道：〝爸爸，你走吧。〞他往车外看了看，说：〝我买几个橘子去。你就在此地，不要走动。〞我看那边月台的栅栏外有几个卖东西的等著顾客。走到那边月台，须穿过铁道，须跳下去又爬上去。父亲是一个胖子，走过去自然要费事些。我本来要去的，他不肯，只好让他去。我看见他戴著黑布小帽，穿着黑布大马褂，深青布棉袍，蹒跚地走到铁道边，慢慢探身下去，尚不大难。可是他穿过铁道，要爬上那边月台，就不容易了。他用两手攀着上面，两脚再向上缩；他肥胖的身子向左微倾，显出努力的样子。这时我看见他的背影，我的泪很快地流下来了。我赶紧拭干了泪，怕他看见，也怕别人看见。我再向外看时，他已抱了朱红的橘子往回走了。过铁道时，他先将桔子散放在地上，自己慢慢爬下，再抱起桔子走。到这边时，我赶紧去搀他。他和我走到车上，将橘子一股脑儿放在我的皮大衣上。于是扑扑衣上的泥土，心里很轻松似的，过一会说：〝我走了，到那边来信！〞我望着他走出去。他走了几步，回过头看见我，说：〝进去吧，里边没人。〞等他的背影混入来来往往的人里，再找不着了，我便进来坐下，我的眼泪又来了。

节录自朱自清 — 背影 —

(a) 现代式的标点

我说道:〝爸爸,你走吧。〞他往车外看了看,说:〝我买几个橘子去。你就在此地,不要走动。〞我看那边月台的栅栏外有几个卖东西的等著顾客。走到那边月台,须穿过铁道,须跳下去又爬上去。父亲是一个胖子,走过去自然要费事些。我本来要去的,他不肯,只好让他去。我看见他戴著黑布小帽,穿着黑布大马褂,深青布棉袍,蹒跚地走到铁道边,慢慢探身下去,尚不大难。可是他穿过铁道,要爬上那边月台,就不容易了。他用两手攀着上面,两脚再向上缩;他肥胖的身子向左微倾,显出努力的样子。这时我看见他的背影,我的泪很快地流下来了。我赶紧拭干了泪,怕他看见,也怕别人看见。我再向外看时,他已抱了朱红的橘子往回走了。过铁道时,他先将桔子散放在地上,自己慢慢爬下,再抱起桔子走。到这边时,我赶紧去搀他。他和我走到车上,将橘子一股脑儿放在我的皮大衣上。于是扑扑衣上的泥土,心里很轻松似的,过一会说:〝我走了,到那边来信!〞我望着他走出去。他走了几步,回过头看见我,说:〝进去吧,里边没人。〞等他的背影混入来来往往的人里,再找不着了,我便进来坐下,我的眼泪又来了。

节录自朱自清 — 背影 —

(b) 复古式的标点

图 1: 两种标点方式

产生的结果为：

```
\newcommand{\manual}{ 手册 }
```

“ 手册” 一词与左右花括号的间距显然过宽。编排本文件时，笔者为了解决这个问题，特别挑选不常用到的中文符号 ⊥，在文件前端添加以下的宏定义：

```
\PUXcatcode`\⊥=\active \def⊥{\kern0pt}
```

然后在“ 手册” 一词的前后端加上 ⊥，如下所示：

```
\begin{verbatim}
    \newcommand{\manual}{⊥手册⊥}
\end{verbatim}
```

这样就可以得到如下所示比较美观的结果：

```
\newcommand{\manual}{手册}
```

这个技巧是利用到以下两个事实：

- verbatim 环境只把所有英文字符的类别改成 other，而没有更动汉字元的类别。
- ⊥ 被定义成 `\kern0pt` 使得 pUTEX 不会添加中英空白（参阅第 13 页）。

例 –30 制作 gbTEX/LATEX 的格式档（format）时，笔者用以下的命令，把 GBK 字符集里的所有符号字符的类别码都设成 12 (other)：

```
\PUXrangecatcode"A1A1 to "A1FE=12
\PUXrangecatcode"A2A1 to "A2FE=12
\PUXrangecatcode"A3A1 to "A3FE=12
\PUXrangecatcode"A4A1 to "A4F3=12
\PUXrangecatcode"A5A1 to "A5F6=12
\PUXrangecatcode"A6A1 to "A6B8=12
\PUXrangecatcode"A6C1 to "A6D8=12
\PUXrangecatcode"A6E0 to "A6EB=12
\PUXrangecatcode"A6EE to "A6F2=12
\PUXrangecatcode"A6F4 to "A6F5=12
\PUXrangecatcode"A7A1 to "A7C1=12
\PUXrangecatcode"A7D1 to "A7F1=12
\PUXrangecatcode"A8A1 to "A8C0=12
\PUXrangecatcode"A8C5 to "A8E9=12
\PUXrangecatcode"A9A4 to "A9F4=12
```

除了上述的类别码命令外，pUTEX 提供另一个区域性参数来控制汉字元类别码：

`\puxCJKcharOther`[=]n

若 $n = 0$（默认），则汉字元的类别码依照前述类别码命令的设置；若 $n \neq 0$，则所有汉字元的类别码一律为 12 (other)。

5.8 中文命令名称

由于正常汉字元默认的类别为 letter，所以你可以用中文来命名命令。譬如：在 TEX 中，若定义如下的中文命令：

```
\def\校名{静宜大学}
```

命令“\校名”即代表“静宜大学”一词。在 LATEX 中，可用如下的方法来定义同样的中文命令：

```
\newcommand{\校名}{静宜大学}
```

你可以将参数 \puxCJKcharOther 的值设成 1 来关闭中文命令的功能（不建议如此做）。

5.9 中文直排

中式直排是利用将汉字逆时针旋转 90 度的技巧来达成。当然，你也必须在 LATEX 中修订纸张的尺寸定义、打印时选择横式（landscape）打印方式，如此才能打印出正确的中文直排效果。你可以在定义字貌时，用设置属性 s =r 的方式（参见 5.2.1 节），来定义旋转字体。不过，PUTEX 提供的参数 \puxgRotateCtext 可以让你更简便地旋转字貌来制作中文直排。它的设置方式如下：

```
\puxgRotateCtext=n
```

若 n 为 0 时，表示不自动旋转字貌。这是 \puxgRotateCtext 默认的参数值。若 n 为任何不等于 0 的整数值时，PUTEX 会自动地将所有非旋转字体的字貌改成旋转字体的字貌，所有旋转字体的字貌改成非旋转字体的字貌。由于 \puxgRotateCtext 是一种旗标（flag）参数，如果你已经将 \puxgRotateCtext 的值设成一个非 0 的数值，你将无法再把它改回 0 值。

6 用于格式化档或 LATEX 套件的命令

这一节介绍的的命令主要是用于格式化档或 LATEX 套件，常规的用户通常不会直接用到它们。如果你属于常规的用户，可以跳过此节的内容。

6.1 设置文件字符集

PUTEX 正蜕变成 CJK (Chinese-Janpenese-Korean) TEX 系统，为了辨识文件所使用的字符集（character set），PUTEX 4 新建底下的命令（区域性）：

```
\puxCharSet[=]n
```

n 是一个整数值，用来代表文件所使用的字符集。当前已定义的字符集如下：

n=0: UCS2 (two-byte Unicode)

n=1: Big5 (Taiwan and Hong Kong)

n=2: GBK (China and Singapore)

n=3: Shift-JIS (Japan)

n=4: KS_C_5601 (Korea)

虽然当前 pTEX 4 的格式档只提供 Big5 和 GBK 的版本，cdi2dvi 也只支持这两种字符集，不过，其他国家人士可以根据本手册所述，制作出来格式档，就可以用 cdi2pdf 来预览和打印。当然，在笔者未完成英文版手册之前，这位仁兄可得先学会阅读繁体中文 :)

未来（pTEX 5?），我们希望能利用这个参数来提供多字符集文件的编排，譬如在 GBK 文件中添加 Big5 编码或 UCS2 编码的文件。

6.2 设置汉字元的型态码

在中文排版的惯例上，句点〝。〞、逗点〝，〞等标点符号应该避免出现在一行的最前端，这些字符称为避首字符。左括弧〝（〞或〝【〞等则应该避免出现在一行的最后端，这些字符称为避尾字符。pTEX 支持这项避行首/行尾字符的排版功能。gbTEX/LATEX 依据微软公司的建议 [1, pp 241–242]，支持避行首 / 行尾字符。这些字符除了涵盖大部分的中文标点符号外，也包含了若干英式标点符号。

为了支持其他字符集的避行首/行尾功能，pTEX 4 新建以下的命令：

`\PUXtypecode` n[=] 0 | 1 | 2

把码值为 n 的汉字元的型态码（typecode）区域性地设成 0（正常）、1（避尾）、或 2（避首）。针对设置一连串字符的型态码，pTEX 提供以下的命令：

`\PUXrangetypecode` m [`to`] n [=] 0 | 1 | 2

把字码从 m 到 n 的字符的型态码都设成 0, 1, 或 2。

例 –1 如果你用以下的命令：

```
\PUXtypecode`\。=0
```

把句点的型态码改为正常，则会取消它的避首功能（即允许句点出现在行首）。

例 –2 制作 gbTEX/LATEX 的格式档时，笔者用以下的命令来设置 GBK 字符集里的避首字符：

```
\PUXrangetypecode"A1A2 to "A1AF=2
\PUXtypecode"A1B1=2    \PUXtypecode"A1B3=2    \PUXtypecode"A1B5=2
\PUXtypecode"A1B7=2    \PUXtypecode"A1B9=2    \PUXtypecode"A1BB=2
\PUXtypecode"A1BD=2    \PUXtypecode"A1BF=2    \PUXtypecode"A1C3=2
\PUXtypecode"A3A1=2    \PUXtypecode"A3A2=2    \PUXtypecode"A3A7=2
\PUXtypecode"A3A9=2    \PUXtypecode"A3AC=2    \PUXtypecode"A3AE=2
\PUXtypecode"A3BA=2    \PUXtypecode"A3BB=2    \PUXtypecode"A3BF=2
\PUXtypecode"A3DD=2    \PUXtypecode"A3E0=2    \PUXtypecode"A3FC=2
\PUXtypecode"A3FD=2    \PUXtypecode"A6E1=2    \PUXtypecode"A6E3=2
\PUXtypecode"A6E5=2    \PUXtypecode"A6E7=2    \PUXtypecode"A6E9=2
\PUXtypecode"A6EB=2    \PUXtypecode"A6EF=2    \PUXtypecode"A6F1=2
```

例 –3　制作 gbTEX/LATEX 的格式档时，笔者用以下的命令来设置 GBK 字符集里的避尾字符：

```
\PUXtypecode"A1AE=1    \PUXtypecode"A1B0=1    \PUXtypecode"A1B2=1
\PUXtypecode"A1B4=1    \PUXtypecode"A1B6=1    \PUXtypecode"A1B8=1
\PUXtypecode"A1BA=1    \PUXtypecode"A1BC=1    \PUXtypecode"A1BE=1
\PUXtypecode"A3A8=1    \PUXtypecode"A3AE=1    \PUXtypecode"A3DB=1
\PUXtypecode"A3FB=1    \PUXtypecode"A6E0=1    \PUXtypecode"A6E2=1
\PUXtypecode"A6E4=1    \PUXtypecode"A6E6=1    \PUXtypecode"A6E8=1
\PUXtypecode"A6EA=1    \PUXtypecode"A6EE=1    \PUXtypecode"A6F0=1
```

6.3 特殊字符表

为了让 PUTEX 的一些命令（如产生中文数字的命令）能够在不同的字符集下正常运作，PUTEX 内部有一个可含 256 个 CJK 字符的特殊字符表（table of local names）。其中，前 128 个（0 – 127）保留给格式档制作者来设置，而且必须按照规定填入适当的字符，才能让某些 PUTEX 命令产生预期的效果；后 128 个（128 – 255）则可让常规用户来运用。底下的命令让你存入一个 CJK 字符到特殊字符表中：

`\PUXlocalnames` n [=] c

其中，n, $0 \le n \le 255$, 是存入位置，c 是存入的 CJK 字符码。譬如底下的命令在特殊字符表中第 54 格中存入 GBK 字符〝甲〞：

```
\PUXlocalnames54="A5D2  % the first char of Heavenly Stems（甲）
```

你可以用 `\the\PUXlocalnames` n 的方式读出第 n 格的字符，譬如：

`\the\PUXlocalnames54` 产生字符〝甲〞

区间	字符	个数	说明
0–9	〇一二三四五六七八九	10	CJK 数字
10–24	〇一二三四五六七八九十百千万亿	15	中文小写数字
25–39	零壹贰参肆伍陆柒捌玖拾佰仟万亿	15	中文大写数字
40–49	０１２３４５６７８９	10	全角阿拉伯数字
50–53	负－廿卅	4	负号与特殊数字
54–63	甲乙丙丁戊己庚辛壬癸	10	天干
64–75	子丑寅卯辰巳午未申酉戌亥	12	地支
76–89	年时分秒日月火水木金土星期曜	14	日期与时间
90–93	春夏秋冬	4	四季

表格 7: 特殊字符表的默认内容

表格 7 列出所有 pUTEX 格式档必须填入的字符与其位置。其中，位置 0–9 的数字字符是供命令 `\PUXcjknumber` 来转换各国当地的数字写法。位置 10–53 的字符则是专门用来产生中文格式的数字（参阅 5.5 节）。〝日月火水木金土〞这样的安排是按照日文星期的顺序。

例 –4 常规的使者用可以使用特殊字符表 128–255 的位置。譬如：若你添加下面的字符：

```
\PUXlocalnames128=‘\①  \PUXlocalnames129=‘\②  \PUXlocalnames130=‘\③
\PUXlocalnames131=‘\④  \PUXlocalnames132=‘\⑤  \PUXlocalnames133=‘\⑥
\PUXlocalnames134=‘\⑦  \PUXlocalnames135=‘\⑧  \PUXlocalnames136=‘\⑨
\PUXlocalnames137=‘\⑩
```

就可以用 5.5 节所介绍的 `\PUXnameseq` 命令把数值 1, 2, ..., 10 映射到①, ②, ..., ⑩：

`\PUXnameseq` n min=1 max=10 offset=128 % $1 \le n \le 10$

6.4 数字处理命令

为了便利其他国家转换数字格式，pUTEX 4 提供底下的命令：

`\PUXsplitnumber` n

运行之后，数字 n 的信息会存入内部暂存器：`\puxnumdigit`, `\puxsign`, 和 `\puxdigit`。譬如，若输入以下的命令：

```
\PUXsplitnumber 39765
number of digits = \the\puxnumdigits
```

```
sign = \the\puxsign          % 1: positive, -1: negative
digit 0 = \the\puxdigit0,    % the right-most digit
digit 1 = \the\puxdigit1,    digit 2 = \the\puxdigit2,
digit 3 = \the\puxdigit3,    digit 4 = \the\puxdigit4,
digit 5 = \the\puxdigit5,    digit 6 = \the\puxdigit6,
digit 7 = \the\puxdigit7,    digit 8 = \the\puxdigit8,
digit 9 = \the\puxdigit9
```

会得到以下的结果：

number of digits = 5
sign = 1
digit 0 = 5, digit 1 = 6, digit 2 = 7, digit 3 = 9, digit 4 = 3,
digit 5 = 0, digit 6 = 0, digit 7 = 0, digit 8 = 0, digit 9 = 0

7 其他 PUTEX 的基本命令

参数 \puxXspace 的用途是为了让 PUTEX 能够正确地处理 LATEX 的 \verb 命令与 verbatim 环境中的空白字符。常规用户可以不用理会它的存在。

你可以在 \end{document} 之前，添加 PUTEX 的 \PUXdumpfontinfo 命令，将文件中所使用的英文字体、汉字貌、汉字型、字体匹配、和字貌匹配等信息输出至 log 文件中。你除了使用这些信息于调试目的外，也可利用其中 TEX 字体的信息来设计字体字貌的匹配规则。底下是 \PUXdumpfontinfo 所输出的部分结果：

```
Tex fonts
0: nullfont dsize= 0.0pt at 0.0pt matched CJK font=5001
1: cmex10 dsize= 10.0pt at 10.0pt matched CJK font=5001
2: line10 dsize= 10.0pt at 10.0pt matched CJK font=5001
...
Chinese faces
0: id=nullcface name=nullcjkface charset=0 weight=400 .....
1: id=default name=default charset=1 weight=400 style=0 w=1.0 h=1.0 d=0.2
2: id=tm name= 细明 charset=1 weight=400 style=0 w=1.0 h=1.0 d=0.2
3: id=mm name= 中明 charset=1 weight=400 style=0 w=1.0 h=1.0 d=0.2
...
Chinese fonts
5001:face= nullcface dsize= 0.0pt at 0.0pt
5002:face= tm dsize= 5.0pt at 5.0pt
5003:face= tm dsize= 7.0pt at 7.0pt
```

```
5004:face= tm dsize= 10.0pt at 10.0pt
...
English/Chinese font faces matching table
0: eface=cmr cface_id=tm cface_num=2
1: eface=cmtt cface_id=tm cface_num=2
2: eface=cmbx cface_id=mb cface_num=11
3: eface=cmti cface_id=mk cface_num=7
4: eface=cmbxti cface_id=mk cface_num=7
```

8 制作中英文索引

PUTEX 的 puidx 程序 当前尚不支持 GBK 码。

9 致谢

笔者除了感谢国科会的资助外，也要特别感谢静宜大学资管系的谢易霖与刘皓朋两位同学。他们撰写了第一版的 cdi2dvi，让 PUTEX 用户能够方便地利用 dvips。

参考文献

[1] N. Kano. *Developing International Software for Windows 95 and Windows NT*. Microsoft Press. 1995. 2nd. Ed. Addison-Wesley. 1995.

[2] F. Mittelbach, M. Goossens. *The LaTeX Companion*. 2nd Ed. Addison-Wesley. 2004.

[3] H. Kopka and P. W. Daly. *A Guide to LaTeX2*. 4th Ed. Addison-Wesley. 2004.

[4] D. E. Knuth. *The TeXBook*. Addison-Wesley. 1986.

[5] L. Lamport. *LaTeX User's Guide and Reference Manual*. 2nd. Ed. Addison-Wesley. 1994.

[6] 蔡奇伟 . *PUTEX* 安装说明 . `http://www.cs.pu.edu.tw/~tsay/putex/install.html`.

A 逻辑 –实体字体映射档

TrueType 汉字体种类繁多，各厂牌字体的命名方式也各不相同。为了使 PUTEX 文件与其 CDI 输出档具有可携性（portability），PUTEX 在内部采用逻辑字体名称而非真实的字体名称， CDI 输出档也记录逻辑字体名称。

当预览、打印、或转换 CDI 档时， cdi2dvi 必须先将逻辑字体转换成真实的字体，才能够产生出正确的字体图像。 cdi2dvi 利用 PUTEX 系统文件 `cfonts.map` 中所定义的逻辑 –实体字体映射关系来做两者之间的转换。cfonts.map 文件的格式非常简单，其规则如下：

- 行中字符 % 之后的文字均视为注解文字。
- 非注解或空白的文字行称为定义行。定义行用来宣告逻辑 –实体字体间的映射关系。
- 定义行由两栏文字所组成，第一栏为逻辑字体的名称，第二栏为实体字体的名称。两栏之间必须以一个或一个以上的空白字符（或 Tab 字符）隔开。
- 逻辑字体名称必须是字貌定义命令 `\PUXcfacedef` 中的中文逻辑字体名称（参见 5.2.1 节）。
- 实体字体的名称必须与 Windows〝控制面板〞中〝字体〞窗口所显示的汉字体名称完全相同。
- 第一个定义行的实体字体是默认的实体字体。当某一逻辑字体无对应的实体字体时， CDI 驱动程序会自动将其映射至此默认的实体字体。
- 每一个逻辑字体只能定义一个实体字体。不过一个实体字体却可能映射至多个逻辑字体。换句话说，逻辑字体与实体字体间的映射关系是一种〝多对一〞的函数。

以下是预设的 `cfonts.map` 内容（由于笔者采用文鼎字体，所以实体字体均为文鼎字体的名称）。

如果你使用别种厂牌字体，请修改其内容。否则将无法产生出正确的字体。

```
default     宋体          % 预设的实体字体
细明        文鼎中明
中明        文鼎中明
粗明        文鼎粗明
特明        文鼎特明
细楷        文鼎细楷
中楷        楷书体
粗楷        文鼎粗楷
细黑        文鼎细黑
中黑        文鼎中黑
```

外中黑	文鼎中黑体外字符集
粗黑	文鼎粗黑
特黑	文鼎特黑
细圆	文鼎细圆
中圆	文鼎中圆
粗圆	文鼎粗圆
中叠圆	文鼎叠圆体
空叠圆	文鼎空叠圆
细隶书	文鼎中隶
中隶书	文鼎中隶
粗隶书	文鼎粗隶
中行书	文鼎中行书
古印体	文鼎古印体
中仿宋	文鼎中仿
粗仿宋	文鼎粗仿
勘亭流	文鼎勘亭流
综艺	文鼎新艺体
POP1	文鼎 POP-4

B 预设的汉字貌

底下是制作 gbTEX 与 gbLATEX 格式档时，所定义的汉字貌与匹配规则：

```
\PUXcfacedef\PUXFbigfive=default default
\PUXsetdefaultcface\PUXFbigfive
\PUXcfacedef\PUXFtm=tm 细明
\PUXcfacedef\PUXFmm=mm 中明
\PUXcfacedef\PUXFbm=bm 粗明
\PUXcfacedef\PUXFsm=sm 特明
\PUXcfacedef\PUXFtk=tk 细楷
\PUXcfacedef\PUXFmk=mk 中楷
\PUXcfacedef\PUXFbk=bk 粗楷
\PUXcfacedef\PUXFsk=sk 特楷
\PUXcfacedef\PUXFtb=tb 细黑
\PUXcfacedef\PUXFmb=mb 中黑
\PUXcfacedef\PUXFbb=bb 粗黑
\PUXcfacedef\PUXFsb=sb 特黑
\PUXcfacedef\PUXFtr=tr 细圆
\PUXcfacedef\PUXFmr=mr 中圆
\PUXcfacedef\PUXFbr=br 粗圆
\PUXcfacedef\PUXFsr=sr 特圆
\PUXcfacedef\PUXFtli=tli 细隶书
\PUXcfacedef\PUXFmli=mli 中隶书
\PUXcfacedef\PUXFbli=bli 粗隶书
\PUXcfacedef\PUXFsli=sli 特隶书
\PUXcfacedef\PUXFtfs=tfs 细仿宋
\PUXcfacedef\PUXFmfs=mfs 中仿宋
\PUXcfacedef\PUXFbfs=bfs 粗仿宋
\PUXcfacedef\PUXFsfs=sfs 特仿宋
\PUXcfacedef\PUXFtsn=tsn 细行书
\PUXcfacedef\PUXFmsn=msn 中行书
\PUXcfacedef\PUXFbsn=bsn 粗行书
\PUXcfacedef\PUXFssn=ssn 特行书
\PUXcfacedef\PUXFtdr=tdr 细叠圆
\PUXcfacedef\PUXFmdr=mdr 中叠圆
\PUXcfacedef\PUXFbdr=bdr 粗叠圆
\PUXcfacedef\PUXFsdr=sdr 特叠圆
\PUXcfacedef\PUXFsdr=edr 空叠圆
\PUXcfacedef\PUXFkd=kd 勘亭流
\PUXcfacedef\PUXFgn=gn 古印
\PUXcfacedef\PUXFje=je 综艺
\PUXcfacedef\PUXFwb=wb 魏碑
\PUXcfacedef\PUXFpo=po POP1
\PUXcfacedef\PUXFyt=yt 颜体
```

```
\PUXcfacedef\PUXFlms=lms 俪中宋
\PUXcfacedef\PUXFgirl=girl 少女
%
% 定义中英文字貌匹配
\PUXfacematch cmr \PUXFtm
\PUXfacematch cmtt \PUXFtm
\PUXfacematch cmbx \PUXFmb
\PUXfacematch cmti \PUXFmk
\PUXfacematch cmbxti \PUXFmk   % 粗斜体用中楷体
```