PUTEX 使用手冊 V2.0 (Rev 1.0)

靜宜大學資訊管理學系

蔡奇偉 副教授

cwtsay@pu.edu.tw

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

中華民國八十八年四月十二日

*本計畫部份是由國科會提供經費贊助完成（計畫編號：NSC-86-2213-E-126-005）。

目錄

1 前言

這份手冊將告訴你如何使用 PUTEX。筆者假設你曾經使用過 TEX 或 LATEX 來排版英文文件（或中文文件）的經驗，因此本文件將不會涉及 TEX 與 LATEX 的使用細節與內部原理。比方說：我們不會告訴你如何產生 $\alpha, \beta, \ldots$ 等的希臘字母，也不會教你如何寫出類似 $\sqrt{x^2+1}$ 的數學公式，更不會談到 glue, box, line breaking 之類的概念。所以，如果你是一位 TEX 新手，筆者建議你先閱讀一些 TEX 或 LATEX 的書籍（如參考書目中的 [1, 3, 4, 5]），具備一些基本觀念之後，再回來細讀此手冊。

PUTEX 構築在 Christian Schenk 先生的 MikTEX 系統上。筆者利用 MikTEX 中的程式發展工具來製作 PUTEX 系統。事實上，PUTEX 本身只包括了修改過的 TEX 程式與幾個輔助程式，其他如 TEX 字型、 LATEX 樣式定義檔、 DVI 驅動程式等，仍然得仰賴 MikTEX 系統的支援。因此安裝 PUTEX 之前，你必須在電腦中先安裝好 MikTEX。

爲什麼取名爲 PUTEX 呢？ 這是因爲筆者服務於靜宜大學，PU 這兩個英文字母是取自靜宜大學英文校名：Providence University 的字頭。因此，PUTEX 應該讀成 P·U·TEX，不過，你也可以將其中的 PUT 合念（如同 put 這個英文字）。

1.1 如何取得 PUTEX?

PUTEX 計畫的 WWW 首頁是：

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

你可以從這個網址下載最新版的 PUTEX 軟體、本手冊的最新版本、與瀏覽 TEX 相關的資訊。由於 PUTEX 必須在 MikTEX 的環境下才能執行，因此若你尚未安裝 MikTEX 的話，也請從上述的網站一併取得 MikTEX。

1.2 PUTEX 簡介

PUTEX 是一套可在中文 Windows 9x/NT 系統下執行的中英文 TEX 排版系統。 PUTEX 改寫 D. E. Knuth 教授 TEX 程式的原始碼，使之能夠直接處理中文。由於不是採用前置處理（preprocessing）的方法來處理 TEX 檔案中的中文， PUTEX 比其他的中文 TEX 系統具有更高度的相容性、擴充性、與彈性。此外，PUTEX 利用 Windows 9x/NT 的中文字型技術來產生中文字型，因而大大地擴充了可運用的中文字型種類。底下是 PUTEX 針對中文排版所設計的功能：

- 可用來排版純中文文件、純英文文件、或中英文混合的文件。
- 支援所有廠牌（如華康、文鼎、全真、或新研澤）、所有字貌（如細明、楷書、行書、甚至勘亭流）的 TrueType 中文字型。

註：PUTEX 並不直接支援英文 TrueType 字型。

- 可使用 TrueType 中文外字集字型。
- 具有標點符號避行首 / 行尾的排版功能，如「。」和「，」等標點符號將不會出現在行首；而「（」和「【」等標點符號也不會出現在行尾。
- 支援以中文來命名巨集指令。
- 可自由調整中文字的字間距（character spacing）。
- 可自由調整中文字與英文字之間的字間距。
- 可自由調整中文字基線（baseline）的位置。
- 支援中文直排的功能。
- 支援以中文數字表示法來顯示整數值的功能，讓你能夠輕鬆地製作出以「一、二、三、···」或以「壹、貳、參、···」作爲標號的條列項目。
- 產生具有可攜性的 CDI（Chinese Device Independent file）檔案。
- 提供 CDI 至 DVI（DeVice Independent file）的轉檔程式，所產生的 DVI 檔可以用 MikTEX的 Yap來預覽列印，或用 dvips 進一步地轉換成 PostScript 檔。
- 提供製作中英文索引的 puidx 程式。

在底下的各節中，我們會進一步地闡明如何使用這些功能。

1.3 PUTEX 的軟、硬體需求

PUTEX 的軟、硬體最低需求如下：

軟體： 中文版的 Windows 9x/NT 4.0（或以上）與 MikTEX。 TrueType 中文字型若干種（請依個人所需安裝）。筆者建議你最少應有細明、楷書、和中黑這三種常用的字型。

硬體： Pentium 90Mz CPU, 16 MB RAM, 50 MB 以上的硬碟空間。

另外，你最好使用 15 吋（或以上）的顯示器，因爲你需要至少 800 × 600 的螢幕解析度，才能夠比較清晰地預覽 DVI 或 CDI 檔。

1.4 安裝 PUTEX

請以 WWW 瀏覽器閱讀檔案 `install.html`，並按照其中所述的步驟來進行安裝。安裝 PUTEX 之後，請務必依照附錄 A 所述的方式來設定 `cfont.map` 這個字體對應檔，讓 PUTEX 能夠正確地利用 Windows 系統中的 TrueType 中文字型。

2 PUTEX 快速入門

PUTEX 的執行方式與英文版的 TEX 系統大致相同，即分爲以下三個步驟：

一、編輯 TEX 或 LATEX 格式的中文文件檔，如 foo.tex。

二、選擇下列適當的指令來產生 CDI 輸出檔：

- 如果 foo.tex 是 TEX 格式，則執行指令 putex foo
- 如果 foo.tex 是 LATEX 格式，則執行指令 pulatex foo

三、利用 cdi2dvi 程式把 CDI 檔轉爲 DVI 檔，然後預覽或列印 DVI 檔。

以下我們就這三個步驟，做進一步的說明：

2.1 編輯中文 TEX 或 LATEX 文件

你可以完全按照標準的格式來編寫純英文的 TEX/LATEX 文件文件。但如果是一份內含中文的 TEX/LATEX 文件，則須注意以下三點，方能順利產生中文的排版輸出：

- 中文字元不可以直接鍵入數學式中，而必須含括在 \mbox（或 \hbox）之中。比方說：

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

  產生的結果爲：$x^2 > 0$ 若 $x \neq 0$。但是以下的輸入方式：

  ```
  $x^2 > 0 若 x \ne 0$
  ```

  則會產生錯誤訊息，而且其結果爲：$x^2 > 0x \neq 0$（其中的「若」字消失不見）。

- 撰寫中文 TEX 的文件時，請在主文件的第一行加上以下的指令：

  ```
  \input cfacedef.tex\rm
  ```

  註：cfacedef.tex 是一個定義中文字貌與中英字貌匹配的檔案。請參見 5.2.1 節與 5.2.2 節。

- 撰寫中文 LATEX 的文件時，請使用第 4 節所述的中文化文件類別（class）：twarticle、twreport、或 twbook。比方說，本文件使用 twarticle，因此第一行是：

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

  若撰寫以英文爲主、中文只佔少部分的文件時，則請使用 twbase 套件（package）。

2.2 編譯中文 TEX 或 LATEX 文件

假定 foo.tex 是一個 TEX 文件檔。你必須在 Windows 9x 的 DOS 視窗中（或 Windows NT 的「命令提示字元」視窗中）鍵入指令：

putex foo （若 foo.tex 是 TEX 格式）

或

pulatex foo （若 foo.tex 是 LATEX 格式）

來進行排版。排版所得的輸出檔，其格式將隨 foo.tex 的內容而異：

- 若 foo.tex 是一個純英文檔，且未使用 cfacedef.tex 或 PULATEX 中文化設定，則輸出檔是一個標準格式的 DVI 檔，並將以 foo.dvi 爲檔名。
- 若 foo.tex 內含中文，並使用了 cfacedef.tex 或 PULATEX 中文化設定，則輸出檔是一個 CDI 格式檔案，並將以 foo.cdi 爲檔名。

由於 CDI 是一套擴充 DVI 格式的檔案規格，其中包含了中文碼與中文字體等的資訊，所以一般的 DVI 驅動程式無法解讀 CDI 檔。你必須用底下兩節所介紹的方式來預覽或列印 CDI 檔。

註：如果你沒有按照上一節所述的規則在中文 LATEX 檔中加入 PULATEX 中文化設定（或在中文 TEX 檔中加入 cfacedef.tex），則輸出檔將以 .dvi 爲副檔名。然而此檔案並非標準的 DVI 檔，也非 CDI 檔格式，因此你將無法預覽或列印其中的內容。

2.3 使用 cdi2dvi 轉檔程式

cdi2dvi 是一個將 CDI 檔轉換成 DVI 檔的工具程式。轉換所得的 DVI 檔可以用 MikTEX 所附的 Yap 來預覽列印、或用 dvips 進一步地將其轉成 PostScript 檔。如果你的印表機不相容於 HP LaserJet 4 系列，則你必須適當地設定 Yap、dvips、與 cdi2dvi，否則 Yap與 dvips 將無法處理 cdi2dvi 所產生的 DVI 檔。設定的方式請參考安裝手冊 [6]。

設定完成以上的程式之後，你只要鍵入以下兩行指令：

```
cdi2dvi foo
yap foo
```

即可在 Yap 的視窗中預覽到排版的結果。由於 Yap具有自動更新的功能，因此你可以保留此視窗在螢幕上，之後只要鍵入

```
cdi2dvi foo
```

無須再執行 Yap程式，即可在 Yap視窗中看到更新的結果。

你可以鍵入以下兩行指令：

```
cdi2dvi foo
dvips foo
```

即可把 foo.cdi 轉成 foo.ps。

cdi2dvi 開始執行時，會先在工作目錄中新建一個名爲 `cdifont` 的子目錄。然後將產生的中文字型檔與 tfm (tex font metric) 檔置於其中。由於 cdi2dvi 每次執行都會先刪除工作目錄中舊有的 `cdifont` 子目錄，因此你不必擔心中文字型檔會日益增加，而佔據了所有的硬碟空間。不過，當你完稿後，不需要在此工作目錄繼續使用 `cdifont` 的話，請記得自行刪除 `cdifont` 目錄（Windows 95/98 可用 `deltree cdifont` 指令，Windows NT 則用 `del/s/q cdifont` 指令）。

cdi2dvi 指令的語法如下：

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

其中，只有 *cdi_file* 不可缺外，其餘的參數與副檔名均可省略。若沒有指定 *dvi_file* 輸出檔，則 DVI 輸出檔將以 *cdi_file*.dvi 爲名。cdi2dvi 的指令選項如下：

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

設定印表機的輸出模式名稱（*mode*）、水平解析度（*x*_resolution）、與垂直解析度（*y*_resolution）。

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

設定螢幕的輸出模式名稱（*mode*）、水平解析度（*x*_resolution）、與垂直解析度（*y*_resolution）。

一般而言，你只要按照安裝手冊 [6] 的步驟設定好 cdi2dvi 之後，就不需要使用上述的指令選項。

3 PUTEX 如何處理空白字元？

空白字元在 TEX 文件中具有語法與語意兩種用途。在語法方面，空白字元用來分隔英文字（words）與分隔語元（tokens）。在語意方面，空白字元用來產生彈性寬度的空白間隔，並作爲可能的斷行點（line-breaking point）。此外，請注意底下兩點 TEX 處理空白字元的方式：

- 除非空白字元被定義成產生空白寬度的指令（如在 \verb 指令或 verbatim 環境中），否則，連續多個空白字元將會被刪減成一個空白字元。
- 跟隨在巨集指令之後的空白字元是用來隔斷指令名稱，因此排版時將全數刪除，而不會產生空白寬度。

PUTEX 爲了彈性與美觀的目的，允許使用者自由調整中文的字間距和中英文的字間距。然而這個功能卻增加了空白字元處理的複雜性，因爲空白字元不再單純地只產生英文字的間距空白，而可能產生以下三種空白間隔：

- **英文空白**：英文字之間的空白間隔
- **中文空白**：中文字之間的空白間隔
- **中英空白**：中文字與英文字之間的空白間隔

當輸入文件時，你只要遵從以下的規則，就能讓 PUTEX 正確地處理大部分的空白字元：

★ 在 \verb 指令與 verbatim 環境中的任何空白字元將會保留而不刪除，並產生英文空白字元的寬度。譬如：

```
\verb| 姓名 ␣␣␣年齡 ␣␣␣住址␣␣␣␣Phone|
```

（␣ 代表空白字元）產生的結果爲：

姓名　 年齡　 住址　 Phone

★ 中文字元之間**不需要**以空白字元隔開。PUTEX 會自動在兩者之間插入**中文空白**。此外，在中文字元間加入一個或一百個空白字元並無差異，因爲它們所產生的排版效果與不加入空白字元完全相同。如果想增加某兩個中文字元的間距，你可以使用 \␣、 \,、或 \hspace 之類的指令。

★ 中文字元與英文字元之間**不需要**以空白字元隔開。PUTEX 會自動在兩者之間插入**中英空白**。此外，在兩者之間加入一個或一百個空白字元並無差異，因爲它們所產生的排版效果與不加入空白字元完全相同。

★ 中文字元與行間（inline）數學式之間**不需要**以空白字元隔開。PUTEX 會自動在兩者之間插入**中英空白**。此外，在兩者之間加入一個或一百個空白字元並無差異，因爲它們所產生的排版效果與不加入空白字元完全相同。譬如底下兩行：

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

產生的結果均爲：

公式 $x^2 = 4$ 的正解爲 $x = 2$。

★ 由於 PUTEX 將非標點中文字元的類別都設定爲 letter，因此你必須以一個以上的空白字元將 TEX 指令與其後的中文字元隔開，否則該中文字元會被視爲指令名稱之一部份，而造成錯誤。比方說：

我愛 \TeX␣的排版能力

產生的結果爲

我愛 TEX 的排版能力

但是

我愛 \TeX 的排版能力

則會讓 PUTEX 誤認「\TeX 的排版能力」爲一個指令，而造成錯誤。

★ 由於 PUTEX 將中文標點字元的類別都設定爲 other，因此當 TEX 指令之後跟著中文標點符號時，你可以省略兩者間的空白。比方說以下三種輸入方式：

「我愛 \TeX。」、「我愛 \TeX␣。」、「我愛 \TeX\␣。」

都會產生相同的結果：「我愛 TEX。」。然而，筆者建議你：

不要在標點符號之前使用空白指令 \␣。

否則該標點符號的「避首 / 尾」功能將失效（參見第 5.1 節）。

★ **請使用空白字元將 TEX/LATEX 指令與前後文字隔開，讓 PUTEX 來決定這些空白的種類。**

★ 若指令前後的空白字元無法產生正確的空白間距時，你可以將其改爲 \␣。通常這樣就可以解決大部分的問題。

★ 若改爲 \␣ 後仍無法獲得正確的空白，你可以用 PUTEX 的指令 \PUXspace（英文空白）、\PUXcspace（中文空白）、或 \PUXcespace（中英空白）來直接設定所需的空白。

★ 你可以在兩個字元間插入 \hbox{} 以制止 PUTEX 在兩者之間加入空白間距。譬如：底下兩行的輸入方式：

我喜歡在 WWW 上搜尋資料
我 \hbox{} 喜 \hbox{} 歡 \hbox{} 在 \hbox{}WWW\hbox{} 上搜尋資料

所產生的結果分別爲：

我喜歡在 WWW 上搜尋資料
我喜歡在WWW上搜尋資料

明顯地，第二行的長度短於第一行的長度。又比方說，twbase 檔提供的 \CDOTS 指令可用來產生「……」。它的定義如下：

```
\newcommand*{\CDOTS}{\mbox{⋯\hbox{}⋯}}
```

底下我們舉一些輸入方式的範例。

例 3–1 如果巨集指令展開後的第一個語元是中文或英文字元（如 \TeX），則你不必在該巨集指令之前鍵入空白字元。譬如：假定指令 \A 的定義為：

```
\newcommand{\A}{ 排版}
```

則底下兩行輸入方式：

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

會產生相同的結果如下：

使用 TEX 來排版真方便！
使用 TEX 來排版真方便！

例 3–2 如果你使用字型設定群組，如 LATEX2.09 的 {\it ...} 或 LATEX 2ε 的 {\itshape ...} 等等，則你不必在其群組之前後鍵入空白字元。譬如底下的兩行輸入方式：

```
\PUTeX␣處理空白的␣{\it 規則一}␣是...
\PUTeX␣處理空白的{\it 規則一} 是...
```

會產生相同的結果如下：

PUTEX 處理空白的規則一是...
PUTEX 處理空白的規則一是...

在少數的狀況下，PUTEX 還是得要靠你的輔助才能正確地處理字元間的空白間隔。底下，我們示範用 \␣ 來取代空白字元␣以解決錯誤空白間隔的問題。

例 3–3 除了「\verb 指令中的第一個字元是中文字元」這個情況外，\verb 指令前後的空白字元都會產生正確的空白間隔。當 \verb 指令中的第一個字元是中文字元時，你必須用 \␣ 來取代 \verb 指令之前的空白字元␣。譬如：

```
我服務於␣\verb|Providence University|␣資管系。
我服務於\␣\verb| 靜宜大學 |␣資管系。
我服務於␣\verb| 靜宜大學 |␣資管系。
```

產生的結果分別為：

我服務於Providence University資管系。
我服務於靜宜大學資管系。
我服務於靜宜大學資管系。

前兩行的結果是正確的；最後一行由於「於」與「靜」之間的空白稍寬，因此不甚美觀。

例 3–4 由於\underline指令是利用數學式的技巧來製作，因此PUTEX會將底線字與前後中文之間的空白視為中英空白。譬如：

你␣\underline{千萬不可以}␣說謊

產生的結果為：

你 千萬不可以 說謊

句中底線字與前後中文字的間距顯然不對。這時你可以用\␣來取代空白字元␣：

你\␣\underline{千萬不可以}\␣說謊

而得到底下的正確結果：

你千萬不可以說謊

例 3–5 假定底線字的首尾是英文字元，則其前後的空白會正確地解釋成英式空白。譬如：

你␣\underline{never}␣說謊

將產生正確的結果：

你 never 說謊

例 3–6 有些指令（如\index）會「暗中」加入一些TEX內部資料結構於其前後字元之間，使得PUTEX無法正確地判斷空白的種類。有時這會造成即使改用前述之\␣的技巧也無法解決。比方說，在本手冊中，筆者定義如下的指令：

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

讓指令\MikTeX除了在本文中顯示出MikTEX以外，也自動地加入索引之中。然而以下兩行的輸入方式：

仍然得仰賴\MikTeX␣系統的支援。
仍然得仰賴\MikTeX\␣系統的支援。

卻會產生如下X與「系」之間的不正確空白間隔：

仍然得仰賴MikTEX系統的支援。
仍然得仰賴MikTEX系統的支援。

因為前者 X 之後的空白被 TeX 所吸收掉，而使得兩字之間並無空白間隔。後者的 \␣ 則被 pTeX 解釋成中式空白。你可以用以下的輸入方式來解決這個問題：

仍然得仰賴 \MikTeX{}␣系統的支援。

其中的 {}（括弧之間不可有空白字元）是用來阻止 TeX 吸收掉其後的空白字元。

例 3–7 有些時候你可以定義巨集指令來避免多打空白字元。譬如有了以下的定義：

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

你就可以打

請參閱 \pgref{...} 之說明

（「之」字前不需空白字元），來取代

請參閱第 \pageref{...}␣頁之說明

（「頁」字前需要空白字元）

4 LaTeX 中文化

爲了讓使用者能夠更方便地利用 LaTeX 來產生中文文件， pLaTeX 提供了以下三種文件類別（document class）：

twarticle	中文化的 LaTeX article 文件類別。
twreport	中文化的 LaTeX report 文件類別。
twbook	中文化的 LaTeX book 文件類別。

此外， pLaTeX 也提供 twbase 套件（package）專供以英文爲主、中文只佔少部份的文件使用。針對某些與 pLaTeX 不相容的套件（如 fancyvrb）， pLaTeX 提供 twpkgpatch 套件來解決不相容的問題。

4.1 twbase 套件

如果你所撰寫的 LaTeX 文件是以英文爲主，中文只佔少部份，那麼你應該採用 twbase 這個套件。其用法如下：

```
\documentclass[options]{class}
\usepackage{twbase}
```

其中的 *class* 可以是任何的 LaTeX 文件類別，如 article、report、book、slide、等等。如果檢視 twbase 檔案的內容，你可以發現到：其中只是讀入檔案 `cfacedef.tex`，以及定義若干巨集指令，而並沒有把英文標題（如 Chapter）換成中文標題。

twbase 裏的巨集指令，也可用於其他的 PuLaTeX 中文化套件與文件類別。以下我們說明這些巨集指令。

4.1.1 PuTeX 的 Logo

twbase 定義了以下三個 Logo 指令：

- `\PUTeX` 產生 PuTeX。
- `\PULaTeX` 產生 PuLaTeX。
- `\PULaTeXe` 產生 PuLaTeX 2_ε。

4.1.2 中式連點

twbase 套件提供 `\CDOTS` 指令，其定義如下：

```
\newcommand*{\CDOTS}{\mbox{⋯\hbox{}⋯}}
```

比方說，若你輸入：

```
鼠、牛、虎、兔、\CDOTS、狗、豬
```

所得之結果將爲：

鼠、牛、虎、兔、⋯⋯、狗、豬

註：請勿用輸入連續兩個「⋯」中文字元的方式來產生上述的中式連點，理由如第 12 頁所述。

4.1.3 中式日期

twbase 套件提供 `\roctoday` 指令。它的作用如同 LaTeX 的 `\today` 指令，用來輸出今天日期，不過 `\roctoday` 會產生類似

中華民國八十七年五月二十日

的中式日期格式。此外，`\Roctoday` 指令則以大寫中文數字來顯示年月日，如：

中華民國捌拾柒年伍月貳拾日

最後，\rocyear 指令可用來產生阿拉伯數字格式的民國年份，如：

今年是民國␣\rocyear␣年

產生的結果是：

今年是民國 88 年

4.1.4 以中文數字顯示 counter 的值

twbase 套件提供的 \cnumber 或 \Cnumber 指令可用來將 counter 的數值以小寫或大寫中文數字的方式顯示。它們的作用有如 LaTeX 的 \arabic 指令。

```
\cnumber{\thesection}     % 以小寫中文數字顯示 counter \thesection 的值
\Cnumber{\thesection}     % 以大寫中文數字顯示 counter \thesection 的值
```

4.1.5 中式編號條列

twbase 套件提供了兩個類似 enumerate 的中式編號條列環境。一個是以小寫中文數字來編號，另一個則是以大寫中文數字來編號。我們用以下的例子來說明它們的用法與所得的結果。

例 4–1 底下是使用「一二三」小寫中文數字編號條列環境的範例。左側是輸入方式，右側則為排版的結果。

輸入行

```
\begin{一二三}
  \item 首先，
    \begin{一二三}
      \item 第一條
        \begin{一二三}
          \item 第一項
            \begin{一二三}
              \item 第一點
            \end{一二三}
        \end{一二三}
    \end{一二三}
  \item 其次，
  \item 最後，
\end{一二三}
```

排版結果

一、首先，

1. 第一條

(a) 第一項

i. 第一點

二、其次，

三、最後，

例 4–2 底下是使用「壹貳參」大寫中文數字編號條列環境的範例。左側是輸入方式，右側則為排版的結果。

輸入行

```
\begin{ 壹貳參 }
  \item 首先，
    \begin{ 壹貳參 }
      \item 第一條
        \begin{ 壹貳參 }
          \item 第一項
            \begin{ 壹貳參 }
              \item 第一點
            \end{ 壹貳參 }
        \end{ 壹貳參 }
    \end{ 壹貳參 }
  \item 其次，
  \item 最後，
\end{ 壹貳參 }
```

排版結果

壹、首先，

1. 第一條

(a) 第一項

i. 第一點

貳、其次，

參、最後，

4.2 標題中文化

除了 twbase 套件以外，twarticle、twreport、和 twbook 文件類別都會把英文標題改成中文。表格 1 列出所更改的標題。其中的第一欄是產生標題的指令名稱，第二欄是標題指令原先定義之英文標題，第三欄則是重新定義之中文標題。若你不滿意任何一個中文標題的名稱，你可以用 `\renewcommand` 指令重新定義之。譬如，若想將預設的中文標題「序」改成「自序」，你只要在文件前端鍵入如下的一行即可：

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

4.3 twarticle 文件類別

`twarticle.cls` 是 LaTeX `article.cls` 的中文化文件類別，因此你可以用下面的方式來使用它：

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

`twarticle.cls` 與 `article.cls` 功能相同，另外也加進了 twbase 套件的內容。

標題指令名稱	原英文標題	中文標題	附註
\prefacename	Preface	序	
\partname	Part	篇	
\contentsname	Contents	目錄	
\listfigurename	List of Figures	圖形目錄	
\listtablename	List of Tables	表格目錄	
\indexname	Index	索引	
\appendixname	Appendix	附錄	
\abstractname	Abstract	摘要	
\tablename	Table	表格	
\figurename	Figure	圖	
\pagename	Page	頁	
\refname	References	參考書目	article
\bibname	Bibliography	參考書目	report & book

表格 1: 中文標題

4.4 twreport 與 twbook 文件類別

twreport 和 twbook 分別是 LaTeX report 和 book 的中文化文件類別，因此你可以用下面的方式來使用它們：

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

或

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

其中的 *options* 稱爲文件類別的選項。由於 twreport 和 twbook 雷同，因此我們就以 twreport 爲例來說明兩者的使用方式。

twreport 除了擁有 report 所有的功能以外，也加進了 twbase 套件的內容。此外，twreport 與 report 的主要差別如下：

- twreport 修改了 \chapter 指令與 \appendix 指令，使前者產生「第一章」之類的章序號，後者產生「附錄 A」之類的附錄序號。
- 目錄裏阿拉伯數字格式的章序號也會改成「第一章」、「第二章」、之類的中式序號。
- twreport 定義了 \twchaptername 巨集指令，其內容是目前的中文章序號，如「第一章」等等。

4.4.1 文件類別的選項

一般而言，twreport 的章序號是使用小寫中文數字，章標題則是靠左對齊。不過，你可以用以下的文件類別選項來改變這些設定：

chapscnum	章序號使用俗體中文數字，如「第廿一章」。
chapucnum	章序號使用大寫中文數字，如「第貳拾壹章」。
chapfcnum	章序號使用正式中文數字，如「第壹拾壹章」。
chapacnum	章序號使用全形阿拉伯中文數字，如「第２１章」。
rightchhd	章標題靠右對齊。
centerchhd	章標題置中對齊。

比方說，以下的文件類別選項設定：

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

會使得章序號改用大寫中文數字，如「第貳拾壹章」，以及使得章標題靠右對齊。

註：目錄及頁首中的章序號也會使用所設定的章序號中文數字格式。

4.4.2 改變章序號

如果想讓 \chapter 產生「第一講」，而非原本的「第一章」之類的章序號，你只要重新定義 \chaptername 這個指令即可（如下一行所示）：

```
\renewcommand*{\chaptername}{講}
```

4.4.3 改變章標題的字型

twreport 內部利用指令 \TWchapHeadingFont 和 \TWchapContentFont 來分別設定章標題及其在目錄中所使用的字型。它們預設的定義如下：

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

使得章標題裏的中文在兩個地方都使用中黑體字型。如果你想改變章標題的中文字型，可以重新定義以上的兩個指令。譬如：

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

將使得章標題的中文改用粗圓體字型。同樣地，以下的定義：

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

將使得在目錄裏的章標題使用比較大的字型，以及中文改用中楷體字型。

註：如果你對以上的 \PUXFbr 與 \PUXFmk 指令感到「霧煞煞」，沒關係，因爲它們是 pTEX 新增用來變更中文字貌的指令。我們會在 5.2 節介紹它們。

4.4.4 中文參照號碼

twreport提供指令 \twref 讓你產生中文參照號碼。比方說，你可以用 \label 指令來定義某一章的標籤，如：

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

然後利用以下的指令：

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

使得 \chapref{makeidx} 產生出如「第八章」而非「第 8 章」的中文參照號碼。

由於指令 \twref 只會將參照號碼的第一個數字改成中文數字，所以應用在節或數學公式時，會得到「八 .1 節」或「公式九 .3」的結果。基於這個限制，指令 \twref 比較適用於如章號或頁碼等只由一個數字所組成的參照號碼。

4.5 twpkgpatch 套件

pLaTeX 只修改了 LaTeX 的 \verb 指令與 verbatim 環境的定義，使其中的空白字元能夠正確地解讀，其他所有的 LaTeX 指令都忠實地保留下來。因此 pLaTeX 與 LaTeX 相容度極高，幾乎所有的 LaTeX 的套件都可以一成不變地使用在 pLaTeX 之中。然而，有些套件提供類似 \verb 指令與 verbatim 的環境，如 fancyvrb 套件，就會與 pLaTeX 不相容。因此 pLaTeX 提供 twpkgpatch 套件來解決這類的問題。

以 fancyvrb 套件爲例，你可以用下面的方式來解決不相容的問題：

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

即載入 fancyvrb 套件之後，再以 fancyvrb 爲選項載入 twpkgpatch 套件。

註：到目前爲止，筆者只發現 fancyvrb 套件與 pLaTeX 不相容。如果你發現到其他不相容的套件時，請通知筆者，我會盡力加以解決。這種狀況應該是極少發生的吧：）

4.6 taiwan 套件還在嗎？

讀到這裏，PUTEX 1.x 版的使用者一定奇怪地問：「taiwan 套件還在嗎？」。這個問題的答案有兩個，好消息是：「它還在」；壞消息是：「它是爲了舊有的文件而存在」。換句話說，筆者自 PUTEX 2.0 版之後不再繼續改良 taiwan 套件。請諸位使用者以後改用前述的 twbase 套件，或 twarticle、 twreport、或 twbook 文件類別。

4.7 一些 LATEX 的中文化技巧

在這一小節中，我們示範一些 LATEX 的中文化技巧。

4.7.1 設定節標題的中文字型

因爲標準的 LATEX 採用 cmbx（粗體字）爲節標題的英文字貌， `cfacedef.tex` 檔又將 cmbx 對應至中黑體，所以節標題的中文字將以中黑體出現。然而，你可以利用 *LATEX Companion* 書中所述的技巧來變更節標題使用的中英字貌 [1, pp 24–26]。

例 4–3 若想用粗圓體取代中黑體以作為節標題的中文字貌，你可以採用以下的設定方式：

```
\newcommand{\seccface}{\PUXFbr}                 % 節標題的中文字貌
\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
```

4.7.2 中式節序號

如果想用中式數字作爲節序號，你可以參考 [1, p.23] 所述的技巧。比方說，以下三個巨集的重新定義：

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

會使得

```
\section{ 簡介 }
\subsection{ 零與一 }
\subsubsection{ 簡史 }
```

產生類似如下的結果：

一　簡介
一.一　零與一
一.一.一　簡史

註：就如 [1] 書中所述，以上的作法並不完全正確。比如說，這些中式節序號在目錄中會造成文字擠在一塊兒的錯誤。在未來版本，我們會提供解決的方案。

4.7.3 設定中式的頁首標題

你可以利用以下的 LaTeX 指令來加入頁首標題：

```
\pagestyle{headings}
```

或利用 fancyhdr 套件[1]（原名爲 `fancyheading.sty`[1, pp 96–98]）來設定中式的頁首標題。譬如底下是一個適用於 twreport 與 twbook 的設定方式：

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

[1] 安裝 PuTeX 時會自動加入 fancyhdr 套件

```
\rhead[\fancyplain{}{\bfseries\leftmark}]%
    {\fancyplain{}{\bfseries\thepage}}
\cfoot{}
```

以上的設定將會產生類似下面所示的頁首標題：

4.7.4 中文的定理標題

你可以仿照下面的定義方式，把定理類（Theorem-like, [3, p. 79]）環境的標題中文化：

```
\newtheorem{thm}{定理}
```

有此定義之後，下列四行的輸入：

```
\begin{thm}
假定 $a$, $b$, $c$, 和 $n$ 均爲正整數。當 $n\geq 3$ 時，$a^n + b^n \neq c^n$
成立。此爲 {\PUXFmb 費瑪最後定理}。
\end{thm}
```

將可得到如下的結果：

定理 1 假定 a, b, c, 和 n 均為正整數。當 $n \geq 3$ 時，$a^n + b^n \neq c^n$ 成立。此為**費瑪最後定理**。

你可以用中文字貌指令來改變定理類環境所使用的中文字貌。譬如：若再定義一個環境，將 thm 內的中文改用 \PUXFmfs（中仿宋）：

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

則

```
\begin{nthm}
假定 $a$, $b$, $c$, 和 $n$ 均爲正整數。當 $n\geq 3$ 時，$a^n + b^n \neq c^n$
成立。此爲 {\PUXFmb 費瑪最後定理}。
\end{nthm}
```

將可得到如下的結果：

定理 2 假定 a, b, c, 和 n 均為正整數。當 $n \geq 3$ 時，$a^n + b^n \neq c^n$ 成立。此為**費瑪最後定理**。

其中大部分的中文均改成了「中仿宋體」。

4.7.5 其他

由於中文字筆畫遠比英文字母來得複雜，因此 LATEX 預設的單行間距往往使得中文文件的頁面過於擁擠。你可以重新定義 \baselinestretch 來增加行距以避免這個問題。筆者建議以下的設定：

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

或

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

最後，本手冊的範例均以反白的例字爲標誌，並以節爲範圍來編號。此範例環境的設定方式如下所示：

```
\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 pUTEX 新增的基本指令與內部參數

針對中文排版的特性，pUTEX 增加了 20 個基本指令和 7 個內部參數。爲了避免與現有 TEX/LATEX 指令的名稱起衝突，我們採用底下的字頭規則來命名它們：

- 指令的名稱均以大寫英文 PUX（代表 pUTEX eXtension）起頭。
- 局部性（local）參數的名稱以小寫英文 pux 起頭。這些參數的作用範圍僅及於設定的群組（group）之內。
- 全域性（global）參數的名稱以小寫英文 puxg 起頭。這些參數的作用範圍涵蓋整個文件範圍。

在表格 2 中，我們分別列出了這些指令與參數的名稱、用途、型態、以及本手冊中參照的章節。其中型態欄中的字母含意如下：

G (Global): 指令或參數具有全域性的影響力。其效力將維持至重新定義爲止。

L (Local): 指令或參數具有局部性的影響力。其效力不會影響外部群組（group）。

D (Don't care): 指令或參數的影響力與範疇無關。

O (Once): 參數僅能變更其值一次。

除此之外，pTEX 也擴充了 TEX 的 \font 指令的語法與功能，使其能夠以類似定義英文字型的方式來定義中文字型。這樣做的目的是爲了讓 pTEX 也能夠使用 LATEX 2ε 的 NFSS2 規格。細節請參見第 5.2.5 節

5.1 避行首 / 行尾標點符號

在中文排版的慣例上，句點「。」、逗點「、」等標點符號應該避免出現在一行的最前端，這些字元稱爲**避首字元**。左括弧「（」或「【」等則應該避免出現在一行的最後端。這些字元稱爲**避尾字元**。pTEX 自 1.2 版起支援這項避行首 / 行尾字元的排版功能。pTEX 依據微軟公司的建議 [2, pp 241–242]，支援表格 3 所列出的避行首 / 行尾字元。此表格除了涵蓋大部分的中文標點符號外，也包含了若干英式標點符號（即其中字碼爲二位數者）。

註：目前的 pTEX 版本不支援使用者自訂避行首/行尾字元的功能，也不允許使用者解除表格 3 中任一字元的避行首/行尾性質。

5.2 中文字貌與字型

與其他的中文 TEX 系統相比，pTEX 提供了最強大但最易使用的中文字型處理技術。它之所以強大的原因在於：你可以使用任何的 Windows 9x/NT 中文 TrueType 字型，而不再受限於少少幾套的中文字型。另外，你也不需要在硬碟中安裝頗佔空間的中文 bitmap 字型。它之所以最易使用的原因在於：你只需使用幾行指令就能在 LATEX 中輕鬆地定義與使用各式各樣的中文字型。

在說明如何設定中文字型之前，我們先區別**字貌**（font face）與**字型**（font）這兩個術語。簡單地說：所謂字貌即字的外觀，而字型等同於字貌再加上字的大小宣告。pTEX 的中文字貌是由底下的屬性來定義：

字體（font look） 區別出字的形狀。以中文爲例，細明體、楷書體、和**中黑體**是三種不同的字體。

字重（font weight） 用以設定字的粗細。中文 TrueType 字型可以定義出 9 種的粗細程度。

字樣（font shape） 用以設定字的樣式，如正體（normal）、斜體（italic）、反白體（reversed）、與旋轉體（rotated）。

名稱	用途	型態	參閱
\PUXcfacedef	定義中文字貌	G	5.2.1 節
\PUXfacematch	匹配中英文字貌	G	5.2.2 節
\PUXfontmatch	匹配中英文字型	G	5.2.6 節
\PUXdumpfontinfo	輸出字型資訊	D	5.9 節
\PUXcfacecspace	調整字貌的中文字間距	G	5.3.2 節
\PUXcfontcspace	調整字型的中文字間距	G	5.3.3 節
\PUXcfacecespace	調整字貌的中英字間距	G	5.3.5 節
\PUXcfontcespace	調整字型的中英字間距	G	5.3.6 節
\PUXspace	插入英文空白間距	D	5.3.7 節
\PUXexspace	插入英文空白間距	D	5.3.7 節
\PUXcspace	插入中文空白間距	D	5.3.7 節
\PUXcespace	插入中英文空白間距	D	5.3.7 節
\PUXcfacedepth	調整中文字貌深度	G	5.4.2 節
\PUXcnumber	轉換成中文數字	D	5.5 節
\PUXscnumber	轉換成中文數字	D	5.5 節
\PUXucnumber	轉換成中文數字	D	5.5 節
\PUXfcnumber	轉換成中文數字	D	5.5 節
\PUXacnumber	轉換成中文數字	D	5.5 節
\PUXchar	輸入中文字元 Big5 碼	D	5.7 節
\PUXchardef	定義中文字元指令	L	5.7 節

(a) pUTEX 新增的指令

名稱	用途	型態	參閱
\puxgCspace	調整所有的中文字間距	G	5.3.1 節
\puxgCEspace	調整所有的中英文字間距	G	5.3.4 節
\puxgCfaceDepth	調整所有的中文字貌深度	G	5.4.1 節
\puxgRotateCtext	逆時鐘旋轉中文字 90 度	O	5.8 節
\puxWcharother	設定中文字元的類性	L	5.6 節
\puxgCdiOut	以 CDI 作爲輸出檔的副檔名	O	5.9 節
\puxXspace	保留空白字元	L	5.9 節

(b) pUTEX 新增的參數

表格 2: pUTEX 新增的指令與參數

21	!	A147	：	A156	–	A16E	》
29	)	A148	？	A157	｜	A170	︾
2C	,	A149	！	A158	—	A172	〉
2E	.	A14A	：	A159	｜	A174	﹀
3A	:	A14B	⋯	A15A	﹍	A176	」
3B	;	A14C	‥	A15B	｛	A178	﹄
3F	?	A14D	，	A15C	﹏	A17A	』
5D	]	A14E	、	A15E	）	A17C	﹂
7D	}	A14F	．	A160	﹀	A17E	）
A141	，	A150	·	A162	｝	A1A2	】
A142	、	A151	；	A164	﹀	A1A4	〕
A143	◦	A152	：	A166	〕	A1A6	’
A144	·	A153	？	A168	﹀	A1A8	”
A145	•	A154	！	A16A	】	A1AA	″
A146	；	A155	｜	A16C	﹋	A1AC	‵

(a) 避行首的字元碼與符號

28	(	A165	〔	A173	﹀	A1A3	〔
5B	[	A167	﹀	A175	「	A1A5	‘
7B	{	A169	【	A177	﹁	A1A7	“
A15D	（	A16B	﹋	A179	『	A1A9	〝
A15F	﹀	A16D	《	A17B	﹁	A1AB	‶
A161	｛	A16F	ﹽ	A17D	（		
A163	﹀	A171	〈	A1A1	｛		

(b) 避行尾的字元碼與符號

表格 3: PUTEX 的避行首 / 尾標點符號

100	特細	200	中細	300	細
400	正常	500	稍粗	600	中粗
700	粗	800	特粗	900	超粗

表格 4: 字重屬性值對應的粗細程度

5.2.1 定義中文字貌

在 PUTEX 中，中文字貌必須用 `\PUXcfacedef` 指令來定義。此指令的語法如下：

`\PUXcfacedef`*faceid* = *ename cname* [*attribute list*]

（等號可以省略），其中各參數的意義如下：

***faceid*:** 代表此中文字貌的指令。你可以用此指令來改變目前使用的中文字貌（參見 5.2.4 節）。

***ename*:** 這一個參數指定中文字貌的英文代名（identifier）。此名稱必須由（大小寫）英文字母組成，而不得使用其他的字元。此外，名稱不得重複使用。

***cname*:** 這一個參數指定邏輯中文字體的名稱（如細明、中楷等）。此名稱最好以中文來命名。你還必須在 cfonts.map 這個檔案中定義其所對應的真實中文字體（參見附錄 A），否則此字貌將使用預設的真實中文字體（通常爲新細明體），而無法列印出你所期望的字體。

***attribute_list*:** 這一個參數選項列指定以下的字貌屬性：

$t = 100\|\cdots\|900$	此屬性設定字貌的粗細（字重）。屬性值所對應的粗細程度如表格 4 所示。若沒有宣告此屬性的話，則字貌預設爲正常的粗細（即 $t = 400$）。
s=i	設定字貌爲斜體字。
s=r	設定逆時針旋轉 90 度的字貌。
s=v	設定爲反白字貌。

這些屬性可以任意組合，也允許以任意的順序宣告。

由於每套 Windows 9x/NT 系統都內附「新細明體」與「標楷體」這兩種字體，因此 PUTEX 內建了底下兩種對應的字貌：

`\PUXcfacedef\PUXFtm=tm` 細明

`\PUXcfacedef\PUXFmk=mk` 中楷

其他的字貌則必須在外部定義。底下我們舉一些定義中文字貌的例子：

細明	tm	中明	mm	粗明	bm	特明	sm
細黑	tb	中黑	mb	粗黑	bb	特黑	sb
細圓	tr	中圓	mr	粗圓	br	特圓	sr
細楷	tk	中楷	mk	粗楷	bk	特楷	sk
細仿宋	tfs	中仿宋	mfs	粗仿宋	bfs	特仿宋	sfs
細隸書	tli	中隸書	mli	粗隸書	bli	特隸書	sli
細行書	tsn	中行書	msn	粗行書	bsn	特行書	ssn
細疊圓	tdr	中疊圓	mdr	粗疊圓	bdr	特疊圓	sdr
勘亭流	kd	古印	gn	綜藝	je	魏碑	wb
POP1	po	顏體	yt	儷中宋	lms	少女	girl

表格 5: `cfacedef.tex` 檔中預先定義的中文字貌

`\PUXcfacedef\a=tm` 細明	錯誤！與內建的字貌 tm 同名
`\PUXcfacedef\b=mk` 中楷	錯誤！與內建的字貌 mk 同名
`\PUXcfacedef\PUXFtmb=tmb` 細明 `t=700`	細明粗體中文字貌
`\PUXcfacedef\PUXFtmi=tmi` 細明 `s=i`	細明斜體中文字貌
`\PUXcfacedef\PUXFtmv=tmv` 細明 `s=v`	細明反白體中文字貌
`\PUXcfacedef\PUXFtmr=tmr` 細明 `s=r`	細明旋轉體中文字貌
`\PUXcfacedef\PUXFmrrv=mrrv` 中圓 `s=r s=v`	中圓旋轉反白體中文字貌

爲了方便起見，PUTEX 在檔案 `cfacedef.tex` 中預先定義了若干常見的中文字貌。這些字貌均採用預設的屬性值。表格 5 列出這些字貌的邏輯中文字體名稱與英文名稱。在 `cfacedef.tex` 中，筆者是以下的兩個規則來命名字貌的英文名稱：

1. 如果字貌集有粗細之別（如細明、中明、粗明、和特明），則第一個字母用 t、m、b、s 這四個字母分別代表細（thin）、中（medium）、粗（bold）、和特（super）四種粗細。如果無粗細之別，則不加此指示粗細的字母。
2. 在代表粗細的字母之後，使用一個或兩個英文字母來代表中文字體，如 m 代表明體、b 代表黑體、k 代表楷書、r 代表圓體、fs 代表仿宋體等等。

此外，字貌指令是以底下的規則來命名：

> 若字貌的英文名稱為 xx，則指令命名為 `\PUXFxx`。

（`PUXF` 中的字母 `F` 代表 Face 之意）。比方說：若英文名稱是 mm 的字貌，其字貌指令則命名爲 `\PUXFmm`。

5.2.2 定義中英字貌的匹配

PUTEX 利用中英字貌匹配的技巧來簡化中文字型的設定方式。比方說：LATEX 英文字貌的預設使用方式爲：cmr 用於正常文字、cmtt 用於定寬字、cmbx 用於粗體字、以及 cmti 用於斜體字。據此，你只要定義以下四個匹配規則：

- 出現在英文字貌 cmr 使用環境的中文，一律使用中文字貌 `\PUXFtm`。
- 出現在英文字貌 cmtt 使用環境的中文，一律使用中文字貌 `\PUXFtm`。
- 出現在英文字貌 cmbx 使用環境的中文，一律使用中文字貌 `\PUXFmb`。
- 出現在英文字貌 cmti 使用環境的中文，一律使用中文字貌 `\PUXFmk`。

就可以輕易地獲得以下的中文字型設定效果：

- 正常的中文字使用細明體。
- 出現在 `\tt` 環境的中文使用細明體。
- 出現在 `\bf` 環境的中文使用中黑體。
- 出現在 `\em` 和 `\it` 環境的中文使用中楷體。
- 改變字型大小的指令（如 `\small` 和 `\Large`）也適用於改變中文字的大小。

上述之中英字貌匹配的規則可以利用 PUTEX 的 `\PUXfacematch` 指令來達成。此指令的格式如下：

`\PUXfacematch` *eface_name* `\`*cface_id*

其中，參數 *eface_name* 是英文字貌的名稱，`\`*cface_id* 是中文字貌指令。其作用將使得出現在英文字貌 *eface_name* 環境中的中文字元，均採用 `\`*cface_id* 所代表的中文字貌，且其字型大小與英文字型相同。比方說：

```
\PUXfacematch cmr \PUXFtm
```

這個指令將英文字貌 cmr 匹配成 `\PUXFtm`（細明）。若中文字元出現在未設定中文字型且未定義匹配的英文字貌環境之中，PUTEX 會自動將其使用的字貌選爲 tm（細明）。

爲了方便使用者製作中文 TEX/LATEX 文件，`cfacedef.tex` 檔中預先定義了下列的中英字貌匹配規則：

```
\PUXfacematch cmr  \PUXFtm    % 正常字爲細明體
\PUXfacematch cmtt \PUXFtm    % 定寬字爲細明體
\PUXfacematch cmbx \PUXFmb    % 粗體字爲中黑體
\PUXfacematch cmti \PUXFmk    % 斜體字爲中楷體
```

這些規則讓使用者可以用字體變化指令（如 \rm, \em, \it, \bf, 等）來選擇中文字型；也可以用字型大小變化指令（如 \small, \large, \huge 等）來改變中文的大小。舉例來說，以上的字貌匹配可以得到下表所示的 LaTeX 文字變化效果：

輸入	列印結果
正常文字 normal	正常文字 normal
{\bf bold（粗體字）}	**bold（粗體字）**
{\it italic（斜體字）}	*italic*（斜體字）
{\em emphasize（強調字）}	*emphasize*（強調字）
{\small 小字}	小字
{\Large 大字}	大字
{\Huge 巨字}	巨字

例 5–1 如果你想把正常中文字改用細圓體，只需將 cmr 的匹配中文字貌改成 \PUXFtr 即可。比方說，你可以在 LaTeX 文件中鍵入：

```
\documentclass...
\usepackage{taiwan}          % taiwan.sty 已自動載入 cfacedef.tex
\PUXfacematch cmr \PUXFtr    % 正常中文字改用細圓體
```

或在 TeX 文件中鍵入：

```
\input cfacedef.tex \rm
\PUXfacematch cmr \PUXFtr    % 正常中文字爲細圓體
```

例 5–2 你可以自行定義其它的中英字貌匹配。比方說：假定你想將 cmbxti（粗斜體英文字貌）與粗黑體中文字貌匹配，你可以在 LaTeX 文件中鍵入：

```
\documentclass...
\usepackage{taiwan}          % taiwan.sty 已自動載入 cfacedef.tex
\PUXfacematch cmbxti \PUXFbb
```

或在 TeX 文件中鍵入：

```
\input cfacedef.tex \rm
\PUXfacematch cmbxti \PUXFbb
```

5.2.3 匹配英文 PostScript 字貌

在這一節中，我們示範如何匹配英文 PostScript 字貌。此處我們假定你熟悉 LaTeX 2_ε 的 NFSS 與 PSNFSS，若非如此的話，請先參考 The LaTeX Companion 書中的說明 [1]。舉例來說，如果打算改用 PostScript Times-Roman 系列的英文字貌來取代 LaTeX 2_ε 預設的 Computer Modern 系列的英文字貌，你可以採用 `times.sty` 套件。此檔案是置於 `\texmf\tex\latex\psnfss` 目錄之中。其主要的內容如下：

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

由於正體字、斜體字、與粗體字是靠 ptm 所產生，所以你可以參考 `ot1ptm.fd` 字型定義檔（非 `t1ptm.fd`），它也是置於`\texmf\tex\latex\psnfss` 目錄之中。由其中的三行定義：

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

可知上述三種字體分別採用 ptmr7t、ptmri7t、與 ptmb7t 字貌名稱，因此你可以加入以下四行：

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

使得「細明」匹配正體英文字、「中楷」匹配斜體英文字、以及「中黑」匹配粗體英文字。

如果你想使用其他的 PostScript 字型套件，請仿照上述的方式找出正體、斜體、與粗體英文字貌名稱，然後再用 `\PUXfacematch` 來設定匹配的中文字貌。

5.2.4 改變目前使用的中文字貌

在 `\PUXcfacedef` 指令中所定義的中文字貌指令（如 `\PUXFmk`）可以用來改變目前群組內的中文字貌（中文字型的大小則依目前的英文字型而定）。字貌指令是一種局部性（local）指令，它的效力涵蓋所處的群組（group）與其內部群組，但不包括外部群組。**在其效力範圍內，中文字貌指令將會取消中英字貌的匹配設定。**

例 5–3 以下的輸入方式：

```
細明體、{\PUXFmk 中楷體}、{\PUXFmb 中黑體}
```

所產生的結果為：

細明體、中楷體、**中黑體**

例 5–4 底下的兩種輸入方式：

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

會產生相同的結果如下：

正常、中楷體、**中楷體**
正常、中楷體、**中楷體**

例 5–5 底下的輸入方式：

```
{\PUXFtm 正常、{\em 強調字}、{\bf 粗體字}}
```

產生的結果為：

正常、強調字、粗體字

而不是：

正常、強調字、**粗體字**

如果想獲得上一行的效果，你必須採用底下的輸入方式：

```
{\PUXFtm 正常、{\PUXFmk 強調字}、{\PUXFmb 粗體字}}
```

5.2.5 定義中文字型

在 TEX 中，我們用 \font 指令來定義一個英文字型。譬如：

```
\font\tenrm cmr10
```

這一行指令定義 \tenrm 爲 cmr10 的字型。大部份的 TEX 字型名稱是由字貌名稱與字型大小所組成。以 cmr10 爲例，cmr 爲其字貌名稱，而 10 表示其字型大小爲 10pt。PUTEX 即根據此觀察來擴充 \font 指令的功能，使之能用於定義中文的字型。其用法如下：

\font*cfontid* CFONT*fn*

其中，*cfontid* 是使用者自訂的中文字型指令名稱；CFONT*fn* 此字則是由三個部份所組成：

1. CFONT 這五個開頭的英文字母是用來告知 pTEX：此處將定義中文字型，而非 TEX 字型。
2. *f* 是一個已定義之中文字貌的英文名稱。
3. *n* 是一個用來宣示字型大小的正整數。

底下是一些中文字型定義的範例（此處採用表格 5 的中文字貌）：

```
\font\a CFONTtm10               % \a 是 10pt 的細明體
\font\b CFONTtm24               % \b 是 24pt 的細明體
\font\c CFONTmk18               % \c 是 18pt 的中楷體
\font\d CFONTmfs12 at 14pt      % \d 是 14pt 的中仿宋體
\font\e CFONTsb12 scaled 2000   % \e 是 24pt 的特黑體
```

如果用 LaTeX 的 \newfont 指令來定義字型的話，以上的例子可改寫成：

```
\newfont{\a}{CFONTtm10}               % \a 是 10pt 的細明體
\newfont{\b}{CFONTtm24}               % \b 是 24pt 的細明體
\newfont{\c}{CFONTmk18}               % \c 是 18pt 的中楷體
\newfont{\d}{CFONTmfs12 at 14pt}      % \d 是 14pt 的中仿宋體
\newfont{\e}{CFONTsb12 scaled 2000}   % \e 是 24pt 的特黑體
```

5.2.6 設定中英文字型的匹配

pTEX 的字貌匹配技術雖然簡化了使用中文字型的步驟，但是完全不考慮字型大小的因素，有時也會帶來一些困擾。pTEX 因而提供 \PUXfontmatch 指令讓使用者可以做更精確細微的字型匹配。 \PUXfontmatch 指令有兩種形式，其語法分別如下：

1. \PUXfontmatch*cfaceid*
2. \PUXfontmatch \efont \cfont

第一種形式是用來匹配目前使用之英文字型。匹配的中文字型將以 *cfaceid* 爲其字貌，並以英文字型的大小爲其大小。第二種形式中的 \efont 是一個英文字型的名稱、\cfont 是一個中文字型的名稱。它的作用是讓出現於 \efont 英文字型環境內的中文採用 \cfont 中文字型。

例 5–6 你可以在 LaTeX 的 preamble 中做以下字型匹配的設定：

```
...
\rm\PUXfontmatch\PUXFmm    % \rm 中文字使用中明體
```

```
\tt\PUXfontmatch\PUXFtm   % \tt 中文字使用細明體
\bf\PUXfontmatch\PUXFmb   % \bf 中文字使用中黑體
\it\PUXfontmatch\PUXFmk   % \it 中文字使用中楷體
\em\PUXfontmatch\PUXFmk   % \em 中文字使用中楷體
\sl\PUXfontmatch\PUXFtr   % \sl 中文字使用細圓體
\rm                       % 回復\rm 字型
\begin{document}
```

讓 LaTeX 的字型變換指令可用來設定其所對應的中文字型。

例 5–7 由於 \PUXfontmatch 是一個全域性的指令，因此以下的設定：

```
{\em 強調 {\em\PUXfontmatch\PUXFtm 強調} \em 強調}
```

所得到的結果為：

強調強調強調 （後兩個「強調」都是細明體）

而非

強調強調強調 （只有中間的「強調」是細明體）

例 5–8 假定你做如下的字型匹配設定：

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

這會使得

```
{\A cmr10 font 和 中楷體 10pt 字型}
```

產生底下的排版結果：

cmr10 font 和中楷體 10pt 字型

其中的中英文字皆為 10pt 大小。

例 5–9 你可以匹配大小不同的中英字型。譬如：

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

將使得

```
{\A cmr10 font 和 中楷體 20pt 字型}
```

產生以下的結果：

{\A cmr10 font 和 中楷體 20pt 字型}

其中的英文字為 10pt 大小，而中文字為 20pt 大小。

5.3 調整文字的間距

pTEX 提供一些參數讓你調整中文的字間距以及中文與英文的間距。調整的對象可以是所有的中文字貌、單一的中文字貌、或單一的中文字型。然而，你應該按照

一、調整所有文字的間距

二、調整單一中文字貌的文字間距

三、調整單一中文字型的文字間距

這個順序下達指令，否則會造成調整無效。比方說：若你先調整某一字型的文字間距，然後再調整該字型所屬字貌的文字間距，則後者的設定將取代前者的設定，而使得前者的設定無效。

所有調整文字間距的指令均具有全域性，設定之後，效果將維持至下一次變更其值爲止。爲了保持中文字的間距協調與統一，筆者建議你最好只在文件的前端設定間距，調整文件整體的緊密程度。若要局部調整，你應該使用 `\hspace`、`\␣`、或 `$\,$` 之類的 LaTeX 指令。

在介紹各個文字間距設定指令之前，筆者必須先說明：

pTEX 間距參數的設定單位是相對於字型大小而非絕對值。

5.3.1 調整所有中文字的間距

pTEX 的參數 `\puxgCspace` 是用來調整所有中文字的間距。它的語法如下：

`\puxgCspace=`n （等號可以省略）

此處的整數值 n 用來設定間距。n 值愈大，則間距就愈寬，反之則愈窄。假定 s 是目前中文字型的大小，則間距的空白寬度（space）、減量（shrink）、與加量（stretch）計算公式如下：

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

\puxgCspace 的預設值是 50。

例 5–10 假定你不滿意 pTEX 預設的中文字間隔，而希望將空白寬度變為字型寬的 1/5，你可以宣告 \puxgCspace=200（因為 200/1000 = 1/5）。

例 5–11 假定你不滿意 pTEX 預設的中文字間隔，而希望僅以中文字型本身的周圍空白來分隔文字，則你可以宣告 \puxgCspace=0。

5.3.2 調整中文字貌的文字間距

pTEX 的 \PUXcfacecspace 指令可用來調整個別中文字貌的字間距。其語法如下：

\PUXcfacecspace*cfaceid*=*n*

其中的 *cfaceid* 是一個中文字貌指令。

例 5–12 由於標楷體的字體比較小，這使得標楷體的文字間距看起來寬了些，因而你可以用指令：

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

將空白寬度設成 −0.05 倍的字型大小，藉此以縮短標楷體文字的間距。

5.3.3 調整中文字型的文字間距

pTEX 的 \PUXcfontcspace 指令可用來調整個別中文字型的字間距。其語法如下：

\PUXcfontcspace*cfontid*=*n*

其中的 *cfontid* 是一個中文字型指令。此外，pTEX 允許你用以下的寫法：

\PUXcfacecspace\font=*n*

來調整目前中文字型的字間距（此處，pTEX 將 TEX 的 \font 指令解讀成目前中文字型，而不是 \font 指令的原意）。

例 5–13 以下的設定：

```
\font\A CFONTmk12
\PUXcfontcspace\A=60
```

將 12pt 中楷體字型 \A 的文字空白寬度設定為 $60/1000 \times 12 = 0.72$pt。

5.3.4 調整中文與英文的字間距

pUTEX 的參數 \puxgCEspace 是用來調整所有中文字與英文字的間距。它的語法如下：

\puxgCespace=n （等號可以省略）

此處的整數值 n 用來設定間距。n 值愈大，則間距就愈寬，反之則欲窄。假定 s 是目前中文字型的大小，則間距的空白寬度（space）、減量（shrink）、與加量（stretch）計算公式如下：

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

\puxgCEspace 的預設值是 150。

例 5–14 假定你不滿意 pUTEX 預設的中英文字的間隔，而希望將空白寬度變為中文字型寬度的 1/5，你可以宣告 \puxgCespace=200。

例 5–15 假定你不滿意 pUTEX 預設的中英文字的間隔，而希望僅以中文字型本身的周圍空白來分隔中英文字，則你可以宣告 \puxgCespace=0。

5.3.5 調整中文字貌的中英文字間距

pUTEX 的 \PUXcfacecespace 指令可用來調整個別中文字貌的中英文字間距。其語法如下：

\PUXcfacecespace*cfaceid*=n

其中的 *cfaceid* 是一個中文字貌指令。

5.3.6 調整中文字型的中英文字間距

pUTEX 的 \PUXcfontcespace 指令可用來調整個別中文字型的中英文字間距。其語法如下：

\PUXcfontcespace*cfontid*=n

其中的 *cfontid* 是一個中文字型指令。此外，pUTEX 允許你用以下的寫法：

\PUXcfacecspace\font=n

來調整目前中文字型的中英文字間距（此處，pUTEX 將 TEX 的 \font 指令解讀成目前中文字型，而不是 \font 指令的原意）。

例 5–16 以下的設定：

```
\font\A CFONTmk12
\PUXcfontcespace\A=60
```

將 12pt 中楷體字型 \A 的中英文空白寬度設定為 $60/1000 \times 12 = 0.72$pt。

5.3.7 插入空白

pUTEX 在某些情況下可能無法產生正確的空白間距，或者你想自行插入某種空白間距，這時你可以用以下的指令來加入所欲的空白間距：

\PUXspace 英文字之間的空白（如同英文字之間的空白 ␣）

\PUXexspace 英文字之間的空白（如同英文字之間的空白指令 \␣）

\PUXcspace 中文字之間的空白

\PUXcespace 中文字與英文字之間的空白

\PUXchar"A140 中文空白字元（參見 5.7 節）

5.4 調整中文字深度

字型深度（depth）定義了其基線（baseline）的位置。由於 TrueType 中文字型的深度通常爲字型大小的 0.2，而 TEX 字型的深度則爲字型大小的 0.25。兩者間的差距有時會造成中英文並置時，高矮不協調。因此 pUTEX 提供參數 \puxgCfaceDepth 讓你調整所有中文字的深度；提供指令 \PUXcfacedepth 讓你調整特定中文字貌的文字深度。調整深度時，你應該先設定 \puxgCfaceDepth的值，然後才用 \PUXcfacedepth 設定個別中文字貌的文字深度。

由於 pUTEX 將深度視爲字貌的屬性，因此你在文件前端設定好字貌深度後，請勿在文件中加以改變，否則會產生不正確的排版結果。

5.4.1 全域性地調整中文字深度

pUTEX 的參數 \puxgCfaceDepth 是用來調整中英文並置對齊的高度。其格式如下：

puxgCfaceDepth=n （等號可以省略）

此處的整數值 n 用來設定深度。n 值愈大，則中文字位置就愈低，反之則愈高。假定字型大小爲 s，深度的計算公式如下：

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

指令	格式	12	123	1230531
\PUXcnumber	小寫	十二	一百二十三	一百二十三萬零五百三十一
\PUXscnumber	俗體	十二	一百廿三	一百廿三萬零五百卅一
\PUXucnumber	大寫	拾貳	壹佰貳拾參	壹佰貳拾參萬零伍佰參拾壹
\PUXfcnumber	正式	壹拾貳	壹佰貳拾參	壹佰貳拾參萬零伍佰參拾壹
\PUXacnumber	阿拉伯	１２	１２３	１２３０５３１

表格 6: 中式數字指令、格式、與範例

\puxgCfaceDepth 參數的預設值是 200（即採用 TrueType 的深度值 0.2）。此設定尚能獲得良好的排版結果。如果你不滿意此預設值，你可以改變它。一般而言，其值的範圍應在 150 至 200 之間。

5.4.2 調整中文字貌深度

PUTEX 的 \PUXcfacedepth 指令可用來調整個別中文字貌的文字深度。其語法如下：

\PUXcfacedepth*cfaceid*=*n*

其中的 *cfaceid* 是一個中文字貌指令。\PUXcfacedepth 是一個全域性的指令。設定中文字貌 *cfaceid* 的深度後，其效果將維持至重新設定 *cfaceid* 的深度爲止。

例 5–17 由於華康隸書體中文與英文並置時，會顯得過高，因此本文件開頭使用深度設定

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

將隸書體的深度調整成 0.25，使其位置下移。

5.5 中式數字

PUTEX 的 \PUXcnumber 指令可用來把一個整數值轉換成中文數字表示法的字串。譬如：\PUXcnumber 123 會把 123 轉換成字串「一百二十三」。PUTEX 提供表格 6 所示的五種不同中文數字表示法。其中，俗體與小寫的差異在於：俗體使用「廿」和「卅」來取代「二十」和「三十」。正式與大寫的差異在於：正式不會省略十進位數字（如表格 6 中數值 12 所示）。阿拉伯則是採用 Big5 的全形阿拉伯數字碼。

例 5–18 以下顯示各式中文數字的產生方法：

輸入	結果
\PUXcnumber 123	一百二十三
\PUXscnumber 123	一百廿三
\PUXucnumber 123	壹佰貳拾參
\PUXfcnumber 123	壹佰貳拾參
\PUXacnumber 123	１２３

5.6 中文指令名稱

PUTEX 允許你以中文來命名指令。譬如：在 TEX 中，你可以定義如下的中文指令：

\def\ 校名　靜宜大學

定義之後，指令「\ 校名」即代表「靜宜大學」一詞。或者，在 LATEX 中，你可以用如下的方法來定義同樣用意的中文指令：

\newcommand{\ 校名 }{ 靜宜大學 }

你可以將參數 \puxWcharother 的值設成 1 來關閉中文指令的功能（不建議如此做）。

5.7 中文字碼指令

PUTEX 仿照 TEX 基本指令 \char 的功能 [4, p.43]，提供 \PUXchar 指令讓使用者能夠用 Big5 碼輸入中文字元。此指令的用法如下：

\PUXchar"hhhh

其中的 hhhh 代表四位的十六進位數字，其值必須大於 255，且爲某一個中文字元的 Big5 碼。

例 **5–19** 由於「佛」字的 Big5 十六進位碼為 A6F2，所以底下的兩種輸入方式會產生相同的排版結果：

佛曰：「不可說」。
\PUXchar"A6F2 曰：「不可說」。

你可以用 \PUXchar 指令來輸入不易從鍵盤鍵入的 Big5 字元，如符號 ♀、♧、①、或日文す、ず、せ等等。

註：有些中文字體（如新細明體和標楷體）並未包含所有的 Big5 符號字元。

例 5–20 正字標記㊣的 Big5 十六進位碼為 A1C0，所以它可以用 \PUXchar"A1C0 的方式產生。

例 5–21 中文空白字元的十六進位 Big5 碼為 A140，所以你可以用 \PUXchar"A140 的方式插入一個中文空白字元。譬如輸入「姓\PUXchar"A140␣名」所得的結果為「姓　名」。

你也可以用 \PUXchar 指令來輸入外字集的中文字元。我們以底下的範例來說明其步驟。

例 5–22 韮菜的「韮」字並不在標準的 Big5 字碼集之內。不過，華康外字集提供此字，且將其字碼定為 8E4D。假定你打算使用華康中黑體外字集中的「韮」字，你可以先做類似以下的定義：

```
\PUXcfacedef\PUXFxmb=xmb 外中黑
\newcommand{\ 九 }{{\PUXFxmb\PUXchar"8E4D }}
```

並在 cfonts.map 檔中加上一行：

```
外中黑      華康中黑體外字集
```

此後，你只要輸入「\九␣菜」就可以產生「韮菜」一詞。

pTEX 也仿照 TEX 基本指令 \chardef 的功能 [4, p. 44]，提供 \PUXchardef 指令。其格式如下：

\PUXchardef*cmd*="hhhh

其作用是將指令 *cmd* 定義成 Big5 碼爲 hhhh 的中文字元。

例 5–23 底下的指令定義：

```
\PUXchardef\正 ="A1C0
\PUXchardef\空 ="A140
```

將可以讓你用指令「\正」來產生㊣、指令「\空」來插入一個中文空白字元「　」。

5.8 中文直排

中式直排是利用將中文字逆時針旋轉 90 度的技巧來達成。當然，你也必須在 LATEX 中修訂紙張的尺寸定義、列印時選擇橫式（landscape）列印方式，如此才能列印出正確的中文直排效果。你可以在定義字貌時，用設定屬性 s =r 的方式（參見 5.2.1 節），來定義旋轉字體。不過，pTEX 提供的參數 \puxgRotateCtext 可以讓你更簡便地旋轉字貌來製作中文直排。它的設定方式如下：

\puxgRotateCtext=*n*

若 n 爲 0 時，表示不自動旋轉字貌。這是 \puxgRotateCtext 預設的參數值。若 n 爲任何不等於 0 的整數值時，PUTEX 會自動地將所有非旋轉字體的字貌改成旋轉字體的字貌，所有旋轉字體的字貌改成非旋轉字體的字貌。由於 \puxgRotateCtext 是一種旗標（flag）參數，如果你已經將 \puxgRotateCtext 的值設成一個非 0 的數值，你將無法再把它改回 0 值。

5.9 其他 PUTEX 的基本指令

如果 foo.tex 是一個純英文的 TEX 文件檔，則 foo.dvi 是一般正常的 DVI 檔案。但是如果 foo.tex 中含有任何中文字元的話，foo.dvi，即使仍以 dvi 爲副檔名，已不再是標準格式的 DVI 檔，這是因爲 PUTEX 會在檔案中加入中文碼與中文字型等的資訊。這使得一般的 DVI 驅動程式無法正確地解讀其內容。爲了避免造成混淆，你可以將 PUTEX 內部參數 \puxgCdiOut 的值設定爲 1，使得輸出檔改以 CDI 作爲副檔名。

註：如果你按照 2.1 節所說的方式加入 cfacedef.tex 或 PULATEX 中文設定的話，你不需要再將參數 \puxgCdiOut 設定爲 1。

參數 \puxXspace 的用途是爲了讓 PUTEX 能夠正確地處理 LATEX 的 \verb 指令與 verbatim 環境中的空白字元。一般使用者可以不用理會它的存在。

你可以在 \end{document} 之前，加入 PUTEX 的 \PUXdumpfontinfo 指令，將文件中所使用的英文字型、中文字貌、中文字型、字型匹配、和字貌匹配等資訊輸出至 log 檔案中。你除了使用這些資訊於偵錯目的外，也可利用其中 TEX 字型的資訊來設計字型字貌的匹配規則。底下是 \PUXdumpfontinfo 所輸出的部分結果：

```
Tex fonts
0: nullfont dsize= 0.0pt at 0.0pt matched Chinese font=257
1: cmex10 dsize= 10.0pt at 10.0pt matched Chinese font=258
2: line10 dsize= 10.0pt at 10.0pt matched Chinese font=257
...
Chinese faces
0: id= tm name= 細明 weight=400 style=0 w=1.0 h=1.0 d=0.169999
1: id= mk name= 中楷 weight=400 style=0 w=1.0 h=1.0 d=0.169999
2: id= mm name= 中明 weight=400 style=0 w=1.0 h=1.0 d=0.169999
...
Chinese fonts
257:face= nullcface dsize= 0.0pt at 0.0pt
258:face= tm dsize= 10.0pt at 10.0pt
259:face= tm dsize= 10.0pt at 10.95pt
...
```

```
English/Chinese font faces matching table
0: eface=cmr cface_id=tm cface_num=0
1: eface=cmtt cface_id=tm cface_num=0
2: eface=cmbx cface_id=mb cface_num=9
3: eface=cmti cface_id=mk cface_num=1
```

6 製作中英文索引

你可以利用 PUTEX 的 puidx 程式來製作中英文索引。puidx 使用的中英文排序法則如下：

- 中文以**筆畫為序**，英文則以**字母為序**。
- 若中文字元與英文字元的字序相同時，則英文字元先於中文字元。
- 中英文合併排序。譬如：A < 一 < B < 丁 < C < 大 < D ⋯ 等等。

根據上述的法則，當中文詞首字元的筆畫序與英文字首的字母序相同時，兩者將置於同一個大項之中（如本手冊的索引所示）。比方說：以一劃中文字開頭的中文詞將與以字母 A（或 a）開頭的英文字置於第一大項之中、以兩劃中文字開頭的中文詞將與以字母 B（或 b）開頭的英文字置於第二大項之中、……、等等依此類推。

puidx 脫胎於 LATEX 的 makeindex 程式，除了底下三點的差異外，保留了其他所有 makeindex 的功能：

- puidx 不支援德文的排序，即指令選項 `-g` 無效。
- puidx 更改 headings_flag 的作用。當 headings_flag 爲正值時，索引中的每一大項將以英文字母與中文筆畫數做爲標題（如本手冊的索引所示）；當 headings_flag 的值爲 0 時，每一大項不具有標題：當 headings_flag 爲負值時，索引中的每一大項只以中文筆畫數做爲標題而不會出現英文字母（適用於製作純中文索引）。
- puidx 能用來製作中英文索引，而 makeindex 無法處理中文索引。

當你參考 makeindex 的使用說明時 [1, 3, 5]，只要記住這三點，就可以完全按照 makeindex 的方式來使用 puidx。

最後，我們介紹 putex 所具有的一項便利功能（同樣適用於 makeindex）。假定 `foo.tex` 是一個中文 LATEX 文件檔。若你將索引的樣式定義存在同目錄中的 `foo.mst` 檔中，則執行以下指令時

```
puidx foo
```

putex 會自動地讀取 `foo.mst` 中的樣式定義。利用這個技巧，你就不需要如 [1] 所述執行以下比較複雜的指令：

```
puidx -s foo.mst foo
```

比方說：本手冊的檔名爲 guide140.tex。底下是 guide140.mst 的內容：

```
preamble "\\begin{theindex}\\small\n"
heading_prefix "{\\bigskip\\large\\bfseries "
heading_suffix "\\medskip}\\nopagebreak\n\n"
headings_flag 1
delim_0 "\\dotfill "
delim_1 "\\dotfill "
delim_2 "\\dotfill "
```

利用以上樣式設定產生出來的索引就如最後兩頁所示。

7 致謝

筆者除了感謝國科會的資助外，也要特別感謝靜宜大學資管系的謝易霖與劉皓朋兩位同學。他們撰寫了 CDI 至 DVI 的轉檔程式 cdi2dvi，讓 PUTEX 使用者能夠方便地利用 dvips。

參考書目

[1] M. Goossens, F. Mittelbach and A. Samarin. *The LaTeX Companion*. Addison-Wesley. 1995.

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

[3] H. Kopka and P. W. Daly. *A Guide to LaTeX2e*. 2nd. Ed. Addison-Wesley. 1995.

[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] 蔡奇偉 . *PUTEX 安裝說明* .
http://www.cs.pu.edu.tw/~tsay/putex/install.html.

A 邏輯–實體字體對應檔

TrueType 中文字體種類繁多，各廠牌字型的命名方式也各不相同。爲了使 PUTEX 文件與其 CDI 輸出檔具有可攜性（portability），PUTEX 在內部採用**邏輯字體**名稱而非真實的字體名稱， CDI 輸出檔也記錄邏輯字體名稱。

當預覽、列印、或轉換 CDI 檔時， CDI 驅動程式必須先將邏輯字體轉換成真實的字體，才能夠產生出正確的字型圖像。 CDI 驅動程式利用 PUTEX 系統檔 `cfonts.map` 中所定義的邏輯–實體字體對應關係來做兩者之間的轉換。cfonts.map 檔案的格式非常簡單，其規則如下：

- 行中字元 % 之後的文字均視爲註解文字。
- 非註解或空白的文字行稱爲定義行。定義行用來宣告邏輯–實體字體間的對應關係。
- 定義行由兩欄文字所組成，第一欄爲邏輯字體的名稱，第二欄爲實體字體的名稱。兩欄之間必須以一個或一個以上的空白字元（或 Tab 字元）隔開。
- 邏輯字型名稱必須是字貌定義指令 `\PUXcfacedef` 中的中文邏輯字體名稱（參見 5.2.1 節）。
- 實體字體的名稱必須與 Windows 9x/NT「控制台」中「字型」視窗所顯示的中文字體名稱**完全相同**。
- 第一個定義行的實體字體是預設的實體字體。當某一邏輯字體無對應的實體字體時， CDI 驅動程式會自動將其對應至此預設的實體字體。
- 每一個邏輯字體只能定義一個實體字體。不過一個實體字體卻可能對應至多個邏輯字體。換句話說，邏輯字型與實體字體間的對應關係是一種「多對一」的函數。

以下是預設的 `cfonts.map` 內容（由於筆者採用華康字型，所以實體字體均爲華康字體的名稱）。

> **如果你使用別種廠牌字體的話，請務必修改其內容。否則將無法產生出正確的字型。**

```
細明      新細明體        %  預設的實體字型爲「新細明體」
中明      華康中明體
粗明      華康粗明體
特明      華康特粗明體
中楷      標楷體
中黑      華康中黑體
粗黑      華康粗黑體
```

特黑 華康特粗黑體
細圓 華康細圓體
粗圓 華康粗圓體
中隸書 華康隸書體
中行書 華康行書體
古印 華康古印體
中仿宋 華康仿宋體 W4
勘亭流 華康勘亭流
綜藝 華康綜藝體
POP1 華康 POP1 體

B 預設的 cfacedef.tex 檔內容

```
%% FILE: cfacedef.tex
%% 此檔案定義一些常見中文字型的邏輯字貌名稱，並將參數 \puxgCdiOut
%% 的值設定爲 1 。此外，定義了以下的中英文字貌匹配：
%%
%%    cmr  => tm
%%    cmtt => tm
%%    cmbx => mb
%%    cmti => mk
%%    cmbxti => mk
%%
%% 不要刪除以下兩行前端的 % 字元。
%%\PUXcfacedef\PUXFtm=tm 細明 （內建細明體字貌）
%%\PUXcfacedef\PUXFmk=mk 中楷 （內建中楷體字貌）
%%
\puxgCdiOut=1\relax
\PUXcfacedef\PUXFmm=mm 中明
\PUXcfacedef\PUXFbm=bm 粗明
\PUXcfacedef\PUXFsm=sm 特明
\PUXcfacedef\PUXFtk=tk 細楷
\PUXcfacedef\PUXFbk=bk 粗楷
\PUXcfacedef\PUXFsk=sk 特楷
\PUXcfacedef\PUXFtb=tb 細黑
\PUXcfacedef\PUXFmb=mb 中黑
\PUXcfacedef\PUXFbb=bb 粗黑
\PUXcfacedef\PUXFsb=sb 特黑
\PUXcfacedef\PUXFtr=tr 細圓
\PUXcfacedef\PUXFmr=mr 中圓
\PUXcfacedef\PUXFbr=br 粗圓
\PUXcfacedef\PUXFsr=sr 特圓
\PUXcfacedef\PUXFtli=tli 細隸書
\PUXcfacedef\PUXFmli=mli 中隸書
\PUXcfacedef\PUXFbli=bli 粗隸書
\PUXcfacedef\PUXFsli=sli 特隸書
\PUXcfacedef\PUXFtfs=tfs 細仿宋
\PUXcfacedef\PUXFmfs=mfs 中仿宋
\PUXcfacedef\PUXFbfs=bfs 粗仿宋
\PUXcfacedef\PUXFsfs=sfs 特仿宋
\PUXcfacedef\PUXFtsn=tsn 細行書
```

```
\PUXcfacedef\PUXFmsn=msn 中行書
\PUXcfacedef\PUXFbsn=bsn 粗行書
\PUXcfacedef\PUXFssn=ssn 特行書
\PUXcfacedef\PUXFtdr=tdr 細疊圓
\PUXcfacedef\PUXFmdr=mdr 中疊圓
\PUXcfacedef\PUXFbdr=bdr 粗疊圓
\PUXcfacedef\PUXFsdr=sdr 特疊圓
\PUXcfacedef\PUXFkd=kd 勘亭流
\PUXcfacedef\PUXFgn=gn 古印
\PUXcfacedef\PUXFje=je 綜藝
\PUXcfacedef\PUXFwb=wb 魏碑
\PUXcfacedef\PUXFpo=po POP1
\PUXcfacedef\PUXFyt=yt 顏體
%
% 定義中英文字貌匹配
\PUXfacematch cmr \PUXFtm      % 正常字用細明體
\PUXfacematch cmtt \PUXFtm     % 電報體用細明體
\PUXfacematch cmbx \PUXFmb     % 粗體用中黑體
\PUXfacematch cmti \PUXFmk     % 斜體用中楷體
\PUXfacematch cmbxti \PUXFmk   % 粗斜體用中楷體
```

C 字型範例

此附錄顯示 pTEX 具備使用各式中文字體的能力。下表中的字體是採用華康天蝶中文字型。

\PUXFtm	細明體 秀麗典雅	細明體	秀麗典雅
\PUXFmb	中黑體 端正沈穩	**中黑體**	**端正沈穩**
\PUXFmfs	仿宋體 卓然俊秀	仿宋體	卓然俊秀
\PUXFmk	楷書體 富潤美麗	楷書體	富潤美麗
\PUXFmm	中明體 工整秀麗	中明體	工整秀麗
\PUXFbm	粗明體 結實雅緻	**粗明體**	**結實雅緻**
\PUXFbb	粗黑體 方正嚴肅	**粗黑體**	**方正嚴肅**
\PUXFmli	隸書體 渾然逸致	隸書體	渾然逸致
\PUXFbr	粗圓體 圓順大方	**粗圓體**	**圓順大方**
\PUXFtr	細圓體 清秀婉約	細圓體	清秀婉約
\PUXFmsn	行書體 行雲流水	行書體	行雲流水
\PUXFkd	勘亭流 日旭東昇	**勘亭流**	**日旭東昇**
\PUXFgn	古印體 古意盎然	古印體	古意盎然
\PUXFje	綜藝體 歡樂無限	**綜藝體**	**歡樂無限**

索引

A・一劃

B・二劃

C・三劃

D・四劃

F・六劃

H・八劃

I・九劃

K・十一劃

L・十二劃

M・十三劃

N・十四劃

O・十五劃

P・十六劃

Q・十七劃

R・十八劃

S・十九劃

T・二十劃

U・二十一劃

V・二十二劃

W・二十三劃

Y・二十五劃