# How to LaTeX

teuton

April 23, 2020

# Revision History

| Revision | Date | Author(s) | Description |
|---|---|---|---|
| 0.6.0 | 2020/4/15 | teuton | 初版文件建立 |
| 0.6.1 | 2020/4/17 | teuton | 改變中文套件使用說明 |
| 0.6.2 | 2020/4/20 | teuton | 新增版本歷史紀錄 |
| 0.7.0 | 2020/4/23 | teuton | 改善原始碼排板 |

# Contents


**1 安裝** **6**

1.1 MS Windows 系統 . . . . . 6

1.2 選個順手的編輯器 . . . . . 6

**2 TeX/LaTeX 語法概說** **7**

2.1 LaTeX 文稿的處理流程 . . . . . 7

2.2 LaTeX 的特殊專用符號 . . . . . 7

2.3 LaTeX 排版上的一些規範或慣例 . . . . . 8

2.3.1 一般性的遊戲規則 . . . . . 9

2.3.2 針對標點符號的遊戲規則 . . . . . 10

2.4 LaTeX 的文稿結構 . . . . . 13

2.4.1 環境（environment） . . . . . 13

2.4.2 最簡單的 LaTeX 的文稿結構 . . . . . 14

2.4.3 preamble 區可以放些什麼? . . . . . 14

2.4.4 章節結構 . . . . . 15

**3 實際上排版玩看看** **17**

3.1 簡單的實例 . . . . . 17

3.1.1 關於換行 . . . . . 18

3.1.2 關於縮排 . . . . . 18

3.2 加入章節標題 . . . . . 19

3.3 加入 title page 資訊 . . . . . 20

3.4 加入目錄（Table of Contents） . . . . . 21

3.5 加入摘要（abstract） . . . . . 21

3.6 加入註解 . . . . . 22
3.6.1 腳註（Footnote） . . . . . 22
3.6.2 邊註（Marginal note） . . . . . 23
3.7 字型的相關調整 . . . . . 24
3.7.1 LaTeX 對字型的屬性描述 . . . . . 24
3.7.2 調整字族、字型系列、字形的指令 . . . . . 26
3.7.3 相對字型大小的調整 . . . . . 28
3.8 原文照列 . . . . . 29
3.8.1 原文照列指令 . . . . . 30
3.8.2 原文照列環境 . . . . . 30
3.9 加入中文 . . . . . 31

**4 空間與位置** **33**
4.1 版面大小 . . . . . 33
4.1.1 紙張大小 . . . . . 33
4.2 調整橫向空間 . . . . . 34
4.2.1 調整橫向空間的環境 . . . . . 34
4.3 條列環境 . . . . . 34
4.3.1 項目式條列環境（itemize） . . . . . 34
4.3.2 列舉式條列環境（enumerate） . . . . . 35
4.3.3 敘述式條列環境（description） . . . . . 36

**5 LaTeX 的標準文稿類別** **37**
5.1 LaTeX 類別的宣告 . . . . . 37
5.2 類別的選擇性參數 . . . . . 38
5.3 類別的種類 . . . . . 39

**6 表格的處理** **40**
6.1 表格的種類 . . . . . 40
6.2 tabbing 環境 . . . . . 41
6.3 tabular 環境 . . . . . 42
6.3.1 tabular 表格的基本結構 . . . . . 42

6.3.2 tabular 環境對欄位的調整 . . . . . 43
6.4 浮動環境 . . . . . 46
6.4.1 基本的浮動環境 . . . . . 47
6.4.2 浮動環境選項參數 . . . . . 47
6.5 圖形的引入 . . . . . 48
6.5.1 引入外來圖檔的方法 . . . . . 48
6.5.2 includegraphics 指令的選項參數 . . . . . 48
6.5.3 指定圖檔的搜尋路徑 . . . . . 50
6.5.4 圖文的旋轉 . . . . . 50

# List of Tables


2.1 有特殊含意的符號與其文稿打法 . . . . . 8
2.2 章節結構表 . . . . . 15

3.1 一些字族簡稱 . . . . . 25
3.2 一些字型系列 . . . . . 25
3.3 一些字形 . . . . . 25
3.4 調整文字指令 . . . . . 26
3.5 相對字型大小 . . . . . 29

4.1 紙張大小 . . . . . 33

5.1 類別的種類 . . . . . 39

6.1 置放位置選項 . . . . . 47

# Chapter 1

# 安裝

使用 TeX/LaTeX 系統，剛開始，比較麻煩的是安裝問題。不過，以現在的作業系統而言，幾乎較流行的作業系統都有現成包好的 TeX 系統套件可以安裝

## 1.1 MS Windows 系統

最常使用的 free 版本，大概就是 MiKTeX 及 texworks
MiKTeX 下載，或是使用這連結:https://miktex.org/download
texworks 下載，或是使用超連結:http://www.tug.org/texworks/
安裝的話都自動化了，應該可以很方便的安裝起來。

## 1.2 選個順手的編輯器

TeX/LaTeX 本身並不內附編輯器，只專注在排版的過程，原始文稿是如何產生的並不插手干涉，當然，有些 TeX/LaTeX 的發行版本，乾脆就弄了個編輯器，把編輯器和排版系統本身連接起來

小編使用的是 WinEdt，這是 shareware ，內建自動選字，但只有 Windows 版本：
https://www.winedt.com/download.html
當然還有更多編輯器可以選擇，如 Vim、GNU Emacs、UltraEdit、Kile 等

# Chapter 2

# TeX/LaTeX 語法概說

這一章要談的是，和一般的純文字文稿及其他 markup 式文件系統在語言上的一般差異性。為了讓觀念上能夠更清楚，以下所述主要是要在命令列執行的，至於編輯器上方便的按鍵及巨集，會在第 3 章開始實際玩看看，請別忘了，到時要再回頭來複習一下這些資料。

## 2.1 LaTeX 文稿的處理流程

使用編輯器選擇編譯格式後編譯，或是利用 latex 這個指令去編譯文稿（通常文稿結尾是.tex）：

latex your.latex

處理中文的話要其他前置處理，這裡暫時先以英文文稿來說明，中文的部份，只要加入中文環境及（或）改用能處理中文的前置處理器就可以了。

這樣經過 latex 處理後，會產生一個 your.dvi 檔，可由 pdflatex 由 your.tex 直接編譯成 PDF 格式的輸出。

## 2.2 LaTeX 的特殊專用符號

在 TeX/LaTeX 的世界，原始文稿都是純文字檔，任何一種編輯器都可以打開來編輯、觀看。而排版指令通常是由反斜線 (\, backslash) 所開頭來引導。註解則是由百分號 (%) 來引導。例如，以編輯器編輯下列文字：

```
. This is my first \LaTeX\ typesetting example.
```

編譯後會變成以下的結果：

This is my first LaTeX typesetting example.

其中的 \LaTeX 就是 LaTeX 的一個指令，會顯示 LaTeX 這個特殊的圖示。

許多現有的符號必須拿來當做控制指令，才能符合排版的多樣化需求。表 2.1 的符號可能都得時時留意，不要未經處理就直接寫進文稿裡頭去。

Table 2.1: 有特殊含意的符號與其文稿打法

| 符號 | 作用 | 文稿上使用 | LaTeX 的替代指令 |
|---|---|---|---|
| \ | 下排版命令 | $\backslash$ | \textbackslash |
| % | 註解 | \% | NA |
| # | 定義巨集 | \# | NA |
| ~ | 產生一個空白 | \~{} | \textasciitilde |
| $ | 進入 (離開) 數學模式 | \$ | \textdollar |
| _ | 數學模式中產生下標字 | _{} | \textunderscore |
| ^ | 數學模式中產生上標字 | \^{} | \textasciicircum |
| { | 標示命令的作用範圍 | \{ | \textbraceleft |
| } | 標示命令的作用範圍 | \} | \textbraceright |
| < | 數學模式中的小於符號 | $<$ | \textless |
| > | 數學模式中的大於符號 | $>$ | \textgreater |
| \| | OT1 編碼 | $\|$ | \textbar |
| & | 表格中的分隔符號 | \& | NA |

(如果文稿要使用這些符號，編輯時使用文稿打法即可)

通常，編輯器的語法顏色會幫助判斷語法是否正確，但不是都能完美無缺，有時還是會漏掉，這時別忘了查看一下 *.log 檔案，例如：編譯 your.tex 檔的話，他的 log 檔就是 your.log。

## 2.3 LaTeX 排版上的一些規範或慣例

除了上面所談到的特殊符號外，也有一些規範或慣例要遵守，有些是比較硬性的規定，有些則只是慣例。

### 2.3.1 一般性的遊戲規則

1. LaTeX 的指令都是大小寫有別的，由 \ 開頭，後接由字母組成的字串或單一的非字母字元。其中由 [ ] 中括號括住的是選擇性參數，可以省略，由 { } 大括號括住的是不能省略的參數，當然，LaTeX 的指令不一定會有參數，但絕大部份都會有參數，只不過把他給省略使用預設值罷了。

2. LaTeX 文稿中，空一個英文空白和空多個英文空白的作用是一樣，LaTeX 會認作一個英文空白。

3. 平常我們編輯純文字檔，按個 *E*nter 鍵，就代表換行，但實際排版出來，一行的寬度是按照排版版面的設定，也就是說，你在文稿中按 *E*nter ，不代表排版後就是從這裡斷行，LaTeX 會依一行應有的寬度經過整體計算後自動補成一整行後再來斷行，而且會在中間自動補足一個空白。這在英文很自然，稱為字 (word) 間空白。

4. 編輯器中，多按幾次 *E*nter 就多空出幾行，但在 LaTeX 文稿裡，多個空白行，和一個空白行是一樣的作用，LaTeX 會把他認作是一個空白行。而這個空白行，LaTeX 同時也會認作是新段落的開始，所以 LaTeX 是以空白行分隔各個段落。

5. LaTeX 預設每個章節的第一個段落的第一行是不內縮 (noindent) ，從第二個段落開始才會內縮 (indent) 。當然，這是可以更改的，見 3.1.2 節。

6. LaTeX 的指令，是從反斜線後第一個字母開始，到第一個非字母符號為止 (包括空白、標點符號及數字) 。因此：

   ```
   This is my first \LaTeX typesetting example.
   ```

   這樣的話，實際結果，因為 \LaTeX 後的空白是屬於指令的一部份，空白將不會被解釋，這樣會印成：

   This is my first LaTeXtypesetting example.

   這種結果，LaTeX 和 typesetting 連在一起了。要避免的話，就要指定指令的作用範圍，例如以下的大括號。或就真的加個空白，例如 \ ，

LaTeX 碰到 \ 就會形成完整的指令，其後的空白就會被真正解釋為空白了：

```
This is my first {\LaTeX} typesetting example.
This is my first \LaTeX{} typesetting example.
This is my first \LaTeX\ typesetting example.
```

所以，正常印出來應該是：

This is my first LaTeX typesetting example.

7. 註解符號（%）

```
This is \LaTeX\ document. Give \LaTeX\ a%
try.
```

這樣一來，排版出來會變成：

This is LaTeX document. Give LaTeX atry.

a 和 try 連在一起了！正常應該是：

This is LaTeX document. Give LaTeX a try.

一般% 用於註解，以協助眾人理解 Latex 程式碼功能及作用；在編輯器中，英文文章按 *E*nter 鍵換行時，尾端加個% ，這樣一來 LaTeX 就不會插入英文字間空白，英文字就可以連成中間沒有空白的一整行了，否則 LaTeX 在整篇文稿斷句時，會自動在原換行處填入一個英文空白。

8. 中英文混合的時候，通常，英文字前後都會留個空白，以便和中文區隔開來，只是這個空白要多大，這就沒有固定的慣例，通常留個英文空白也是可以，要講究的話，等談到中文排版相關議題時再來討論，目前就養成習慣，英文單字前後留個英文空白。

### 2.3.2 針對標點符號的遊戲規則

1. 中英文的引號不一樣，這裡請特別注意，許多人常常搞錯。中、英文引號不管單雙都要分左右。英文的話，左邊的引號是 grave accent ，是鍵盤左上方 *E*sc 或 *F*1 下方有波形號的那一個鍵；右邊的是 apostrophe ，也就是 *E*nter 鍵隔壁的那個鍵。雙引號的情形是鍵入兩次的左單引號及兩次的右單引號，而不是用" 這個一次完成兩個點的

ditto marks 。所以，實際上在鍵入文稿時是：

```
Please press an `Esc' key.
Please press an 'Esc' key. 這是錯誤示範！
``This sentence.''
"This sentence." 這是錯誤示範！
```

排版出來的情形是：

Please press an ‘Esc’ key.

Please press an ’Esc’ key. 這是錯誤示範！

“This sentence.”

”This sentence.” 這是錯誤示範！

中文的話，我們是使用中文全形的「、」及『、』，在中國大陸則已改用和英文相同形狀的全形符號，但這在中文直排時會出問題，因此，中文的單、雙引號還是得維持我們目前使用的。

2. LaTeX 會在英文文章的一個句子結束和另一個句子開始的中間，自動調整成較大一點的空白，這可以增加文章的易讀性。所謂一個句子結束，例如：句點 (.) 、問號 (?) 、驚嘆號 (!) 及冒號 (:) ，這當然是指英文的半形標點符號，不是中文的全形標點符號。你可以注意一下上面所舉的 例子 ，在 document. 和 Give 之間的空白會稍微大於其他英文單字間的空白。

   現在的問題是，如果這些標點符號後面不是另一個句子的開始的時候，LaTeX 無法去判斷這種情形，這時得由我們自己自行判斷、處理了。例如英文縮寫字：

   ```
   I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name.
   I am Mr.~Edward G.J. Lee, G.J. is a abbreviation of my name.
   I am Mr.\ Edward G.J. Lee, G.J. is a abbreviation of my name.
   ```

   其中 Mr.\ Edward 的寫法，和 Mr.~Edward 幾乎是一樣的，都是強迫插入一個比較小的正常單字間空白，差別在於後者也另外表示不可以從這裡換行，通常用在人名的時候，讓他們不致中斷，一般在人名的排版，包括他的頭銜、職稱，是不中斷成兩行而分開的。而且整個文句較長的話，以後者較恰當，才不會因為斷行被分成兩半，這個 ~ 符號也因此在 TeX 的專有名詞，就稱為 tie，把他們綁住的意思。排版

出來的時候會變成：

I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name.

I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name.

請放大去仔細比較一下結果就知道了。第二行的才是正確的，Mr. 和 Edward 之間的空白是正常單字間空白，比第一行的句子結束空白要小一點點。其他有使用到縮寫字的場合，例如：‘Dr.’、‘etc.’、‘e.g.’、‘i.e .’、‘vs.’、‘Fig.’、‘cf.’、‘Mrs.’，這些都不是代表句子結束，所以，要插入一個正常空白。

G.J. 後面沒有插入正常空白因為，J 是大寫的，這時 LaTeX 不會去誤認為是句子結束，通常句子結束時的句點前的那個字母是小寫的。

3. 刪節號中文英也是不同，英文是三點，如果碰到句點的話，則是四點。中文的話是六點，碰到中文句點很容易就分得清楚。但是英文這個三點，不是就打個三個句點了事，這樣的點太密集，可以使用 \ldots 或 \dots 指令，例如：

```
I'm not a good man ..., but a good husband .... 錯誤示範！
I'm not a good \ldots\ man \ldots, but a good husband \ldots.
I'm not a good \dots\ man \dots, but a good husband \dots.
```

排版出的來結果是：

I’m not a good ... man ..., but a good husband .... 錯誤示範！

I’m not a good … man …, but a good husband ….

I’m not a good … man …, but a good husband ….

4. 破折號。在英文，相當於破折號的可能有三種：

   - hyphen
     這是最短的 dash ，通常就是鍵入 - 就行了，例如 father-in-low，這樣會表現成 fater-in-low 。
   - en-dash
     這是最常用的破折號，是鍵入兩個 hyphen 。例如 1991--2003 年，這會表現成 1991–2003 年。
   - em-dash

這是最長的 dash，由三個連續的 hyphen 組成，應該是最相近於我們中文所說的破折號。例如 `I am---a good man.` 會表現成 I am—a good man. 。

- 真正的減號
  這應不能算是破折號，而是實際的減號或負號，這要進入數學模式，例如負五要寫成 `$-5$` ，表現出來是 $-5$ 。不能直接鍵入一般的負號那個鍵來充數，這是因為 TeX/LaTeX 的數學式子的用字和間隔處理，和一般內文不同的關係。

5. 避頭點
   這可是排版的重要功能。英文的通常沒有問題，LaTeX 會自動避開處理，中文標點符號，除了破折號及刪節號，沒有開口的 (逗號句號問號等等)，不能置於最開頭，開口向右的 (引號括號)，不能置於最右，開口向左的，不能置於最左。

## 2.4 LaTeX 的文稿結構

### 2.4.1 環境（environment）

上一節所談的都是指令，雖然也可以由大括號來定作用範圍，但如果是一整段，甚至是一整篇文章都要作用時，那指令可能就不很適合了，因此，LaTeX 也有一種巨集結構，稱為環境（environment），主要是讓作用範圍能擴大至較大的範圍。

所有的環境起於 `\begin{環境名稱}` ，止於 `\end{環境名稱}` ，這兩個指令之間的文稿都會被作用，而且，環境之內還可以套用其他不同的環境。

LaTeX 文稿的內文，其實就是包在一個 `\begin{document}` 和 `\end{document}` 這個 document 環境當中。

### 2.4.2 最簡單的 LaTeX 的文稿結構

以下就是所有 LaTeX 必需具備的文稿大結構：

```
\documentclass{report}
    這裡是 preamble 區
\begin{document}
    這裡是本文區
\end{document}
```

\documentclass{report}告訴 LaTeX 使用哪一種類別，目前使用的是 report 類別。preamble 區，則是下一些會影響整個文稿的指令，及引用巨集套件的地方，完全不引用巨集，也不使用影響全文的指令的話，preamble 區就是空白，不寫任何東西。本文區，就是我們實際上寫文章的地方。

### 2.4.3 preamble 區可以放些什麼?

這裡可以引用巨集，而且會影響整篇文稿的指令，例如一些事先定義好的指令，想在整篇文稿中使用，就可以置放在 preamble 區。

#### 2.4.3.1 巨集的引用

本文主要是標準 LaTeX，但前面已提到，會有些巨集套件不得不要引用，底下就來說明如格引用套件。這些套件都是一般 TeX 系統都會附上的。

指令及環境要如何開頭都介紹過了，現在來看看引用巨集要怎麼開頭。

```
\documentclass{report}
\usepackage{color}
\begin{document}
\textcolor{blue}{This is blue color.}
\end{document}
```

編譯一下，看看結果是什麼？這裡使用的就是 color package，裡頭是由 TeX/LaTeX macro 所寫成一個巨集套件。一般簡單的我們就稱為巨集(macro)，複雜一點的就稱為巨集套件 (package)，其實，裡頭都是一樣的，只不過大小及有沒有整理成一個系統的差別。

#### 2.4.3.2 影響整篇文稿的指令

會影響整篇文稿的指令，通常也是放在 preamble 區，例如：

```
\linespread{1.36}
\parindent=0pt
```

\linespread 是在控制上下行的行距，這裡就是將行距變成原來的 1.36 倍。行距呢是這一行的基線（baseline）到下一行的基線的距離，通常英文文章不必去調整他的行距，但中文得適當加大行距以利閱讀。

\parindent 是調整段落內縮的程度，這裡調整成 0，也就是說各段落都不內縮的意思，也可以調整成其他的值，LaTeX 就會依這個值去內縮。當然，不去設定的話，LaTeX 就會依他的預設值去內縮。

### 2.4.4 章節結構

本文區當然是我們寫文章的主要地方，及一些微調。在 LaTeX 的文稿裡頭，章節標題的形成都是由同樣的指令來控制的，這樣有一個好處，臨時插入章節標題及其內文時，我們不必去理會標題編號及目錄的問題，也不必去理會要用什麼字型、及字型大小要多大，LaTeX 會自動計算處理，字型大小也會和內文使用的字型大小互相配合調整，使用者就專心在內文構思、寫作即可。以下由列表2.2 來瞭解整個章節結構：

| 深度編號 | 指令 | 作用及注意事項 |
|---|---|---|
| -1 | \part{} | 最大的結構，中文通常稱為「部」。 |
| 0 | \chapter{} | 章。在 article 類別裡頭沒有章。 |
| 1 | \section{} | 節。 |
| 2 | \subsection{} | 小節。 |
| 3 | \subsubsection{} | 次小節。 |
| 4 | \paragraph{} | 段落。 |
| 5 | \subparagraph{} | 小段落。 |

Table 2.2: 章節結構表

章節標題的內容就是直接寫入指令的大括號裡頭就可以了，LaTeX 在排版時會自動使用粗體、加入章節編號及納入目錄裡頭。

至於第一欄的深度標號（secnumdepth），book/report 類別的深度標號是 2，article 的是 3。就是說 book/report 類別的文稿，在 \subsection{} 以後（subsection 本身仍會編號），章節就不再編號了；同樣的，在 article 類別的文稿，在 \subsubsection{} 以後就不編號了。但仍然會獨立出一單獨行來表示這個是標題。不編號了的章節內容，當然也就不納入目錄裡頭了。這當然是可以更改的，像這篇文章，在 preamble 區就有一個設定：

```
% let the depth of report to subsubsection
\setcounter{secnumdepth}{3}
```

所以，這篇文章雖然使用的是 report 類別，但是章節的深度標號是標在 3，也就是說會編號到 subsubsection 為止，但這仍然是沒有編入目錄中的。指令不必去死記，只要知道有個這樣功能的指令就夠了。

# Chapter 3

# 實際上排版玩看看

本章主要是簡單的實例說明，先進入狀況再談其他。先來個「最高指導原則」：學會控制空間，你就學會排版了！

## 3.1 簡單的實例

這裡就把前一章所談到的一些內容整理成一個文稿，先來試試看，這裡先使用 report 類別文稿，因為 article 類別文稿是沒有 chapter 的：

```
% example1.tex
\documentclass{report}
\begin{document}
This is my first {\LaTeX} typesetting example.\\
This is my first \LaTeX{} typesetting example.\\
This is my first \LaTeX\ typesetting example.\\
I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name.\\
I am Mr.\ Edward G.J. Lee, G.J. is a abbreviation of my name.\\
Please see Appendix A. We will be there soon.\\
Please see Appendix A\null. We will be there soon.
\end{document}
```

使用編輯器編輯，然後存檔成 exmaple1.tex，這樣就可以編譯了

### 3.1.1 關於換行

每行最後加了個 \\，這表示強迫換行的意思，否則 LaTeX 會依版面預設的寬度來換行，就不會是一個句子一行了，試著把這個 \\ 拿掉，再來編譯試看看結果，就會知道怎麼一回事了。也可以使用 \newline 這個指令，當然，我們都會聰明的選用較短的指令。而且 \\ 可以控制換行時的間隔，這在 \newline 則不行。例如：

```
Please see Appendix A. We will be there soon.\\[1cm]
Please see Appendix A\null. We will be there soon.
```

這樣的話，兩行之間的行距就是原來的行距再加上 1cm。甚至，也可以是負數的參數，這樣行距就會變成原來的行距減去 1cm，當然，如果設過頭了的話，兩行可能會重疊在一起。既然，這裡使用的是方括號，表示這些參數是可以省略的。

另外，\linebreak[n] 也可以強迫換行，n 代表由 1–4 的建議值，數值愈大表示愈是強烈建議，不設定的話，就是換或不換兩種選擇，沒有中間地帶。和前面所說的不同處是，這種換行會把原來那一行句子的長度平均布滿版面上行寬的長度。例如：

```
Please see Appendix A. We will be there soon.\linebreak
Please see Appendix A\null. We will be there soon.
```

排版後會表現成：

Please see Appendix A. We will be there soon.
Please see Appendix A. We will be there soon.

### 3.1.2 關於縮排

第一行縮排了！這是因為完全沒有分章節，所以，LaTeX 就把這些內容當做是引言的部份，依 LaTeX 的安排，引言開頭是會縮排的。要解決這個問題，可有兩種方法：

1. 在第一行之前加入 \noindent 來指示 LaTeX 不要去縮排。但是這只作用在下指令的地方，其他該縮排的地方還是會縮排。

2. 在 preamble 區加入 \parindent=0pt，這表示讓全文的縮排為 0pt，當然，這就表示全文都不要縮排了。

## 3.2 加入章節標題

在 LaTeX 裡頭，要加入章節標題實在是太容易了，也不必去管字體的大小及置放的位置，盡管加上去就對了！LaTeX 會替我們安排一切。我們這裡仍然以 report 類別來說明，因為 article 類別裡頭，沒有章，只能適用於較簡單的短文。

```
% example2.tex
\documentclass{report}
\begin{document}
This is the first experience of \LaTeX.
\chapter{Checkpoint city!}
\section{It's a beautiful city}

Progress city!

Right?

\section{that checkpoint city!}

You can all

``have'' some

\chapter{second checkpoint}

It's Christmas

burny stuff
\end{document}
```

請注意他什麼時候會縮排，什麼時候會換頁。report 類別，新的一章會換頁，如果想節省一點空間，可以換用 article 類別，\chapter{} 改用 \section{}，原來 \section{} 就改用 \subsection{}，這樣就不會換頁，

內容就會連續下去了。可以試著把 report 改成 article 及 book 再重新編譯一次，試試看結果有何不同。

## 3.3 加入 title page 資訊

這是指內頁的第一頁，在 LaTeX 裡頭，我們就稱為 title page。在 LaTeX 的標準格式裡，他包括了標題（title）、作者名字（author）、日期（date）及感謝詞（thanks）。要注意的是，在 report/book 類別，title page 是自成一單獨頁的，但在 article 類別裡，他是和本文連起來的。我們就以上面的文章為例，要修改的地方是 preamble 區及本文區的 \maketitle：

```
% example3.tex
\documentclass{report}
\title{It's a troll level}
\author{DGR\thanks{Thanks to the creator.}
          \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
This is the first experience of \LaTeX.
\chapter{Checkpoint city!}
\section{It's a beautiful city}
...
```

我們可以發現，這一頁是不編頁碼的，從下一頁開始才是第一頁。作者可以有多個，使用 \and 指令來連接。日期不一定要有，如果沒有 \date{\today} 這個指令，那還是有日期，但只能固定在今天。如果內容過長，他會自動折行，但也可以手動加 \\ 來強迫換行，不管如何換行，整個句子是居中排列的。\maketitle 是下在本文區的開頭，如果不下這個指令，那編譯時不會有什麼錯誤，只是就沒有 title page 了。

## 3.4 加入目錄（Table of Contents）

加入目錄（Table of Contents）對 LaTeX 而言，更是輕而易舉的事情，只要在本文開頭加個 \tableofcontents ！依上面的例子，修改成：

```
% example4.tex
\documentclass{report}
\title{It's a troll level}
\author{DGR\thanks{Thanks to the creator.}
        \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
This is the first experience of \LaTeX. % 此時這裡沒有作用
\chapter{Checkpoint city!}
\section{It's a beautiful city}
...
```

這裡千萬要注意的是，\tableofcontents 要加在 \maketitle 的後面，否則目錄會印在 title page 之前。而且要編譯兩次。第一次產生 example4.toc，然後第二次編譯再跟據這個 toc 檔，真正編入目錄。

## 3.5 加入摘要（abstract）

這不一定會有，如果要加入的話，可使用 abstract 環境，在這個環境中的文章，左右會縮排。要注意的是，只有 article/report 類別才有 abstract，book 類別不能使用這個環境。

```
% example5.tex
\documentclass{report}
\title{It's a troll level}
\author{DGR\thanks{Thanks to the creator.}
        \and Nobody\thanks{Thanks to nobody.}}
```

```
\date{\today}
\begin{document}
\maketitle
\begin{abstract}
It will be a lot of fun.
\end{abstract}
\tableofcontents
\chapter{Checkpoint city!}
\section{It's a beautiful city}
...
```

report 類別的摘要自成一頁，不編頁碼，且不會編入目錄中，這和一般的論文格式可能會不一樣，使用時請注意。article 的類別則仍然是和本文相連的，會出現在文章標題之後。

abstract 和 summary 在較正式的論文是有區分的，通常 abstract 在文前；summary 則在文後。但目前一般性的文章則沒有這樣區別，通通當成「摘要」。通常，摘要裡頭是不用註解、無交互參照也不使用公式圖表的。

## 3.6 加入註解

在 LaTeX 裡頭，註解可有兩種方式，一種是腳註（footnote），一種是邊註（marginal note）。通常 LaTeX 的腳註預設是由阿拉伯數字在編號，置於頁底部。在沒有部（part）的情形下，report/book 類別，編號每章會從頭起算，article 類別則會連續，而且，會使用 footnotesize 的字體印出。邊註則不編號，字體是正常大小。

### 3.6.1 腳註（Footnote）

在所要加註的那個字後，使用 `\footnote{}` 指令即可，解說的文字就寫入大括號之內，一般 LaTeX 的指令在此都仍然有有作用，會印在此頁的底部，以小一點的字來印出，並加上編號。以下我們就試試看在 Right 這個字來做腳註。請注意，Right 這個字和 `\footnote{}` 之間是沒有空白的。

```
% example6.tex
```

```
\documentclass{report}
\title{It's a troll level}
\author{DGR\thanks{Thanks to the creator.}
        \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
\chapter{Checkpoint city!}
\section{It's a beautiful city}

Progress city!

Right\footnote{Of Course!}?
...
```

### 3.6.2 邊註（Marginal note）

邊註只是把 \footnote{} 換成 \marginpar{} 而已，內容仍然寫入大括號內。但和腳註不一樣的是，他沒有編號（因為就在旁邊，無此必要），他的字體也不會小一號，和內文的字體大小是一樣的，這在後面討論到字型的時候會談到如何改變字體的大小。

```
% example7.tex
\documentclass{report}
\title{It's a troll level}
\author{DGR\thanks{Thanks to the creator.}
        \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
\chapter{Checkpoint city!}
```

```
\section{It's a beautiful city}

Progress city!

Right\marginpar{Of Course!}?
...
```

## 3.7 字型的相關調整

TeX/LaTeX 的字型系統算是相當複雜的，這裡不多談其中原理，站在使用者的角度，我們只要知道怎麼使用就行了。在這裡，我們說字型（font），指的是字型本身的一個總稱，或稱為字體，在字的形狀的時候，我們就稱為字形（font shape）。

### 3.7.1 LaTeX 對字型的屬性描述

在 LaTeX 裡，對於字型的描述，使用了五種屬性來說明，這五種屬性，也是 LaTeX 巨集中常要使用到的參數，甚至是錯誤訊息標示字型來源的時候，會把字型的這些屬性給顯示出來。

1. 字型編碼（font encoding）
   這裡所謂的字型編碼，指的是各個個別的字在一個字型裡頭的排列順序及安排方式。原始的 TeX 字型編碼我們就稱為 OT1(Old TeX text encoding)，這是預設的，如果都不指定字型編碼，那所使用的就是 OT1 編碼。

2. 字族（font family）
   指同一設計類型的字型集合的名稱，例如羅馬字族（roman）、打字機字族（typewriter）等等，通常前面會冠上製作商或製作人的名稱，例如 Knuth 教授設計的，稱為 'Computer Modern Roman'，Adobe 公司製作的羅馬字族稱為 'Adobe Times'。我們預設使用的，當然就是 Knuth 教授所設計的 Computer Modern fonts。表3.1 為一些例子：

Table 3.1: 一些字族簡稱

| 簡稱 | 代表意義 |
|---|---|
| cmr | Computer Modern Roman |
| cmss | Computer Modern Sans Serif |
| cmtt | Computer Modern Typewriter |

3. 字型系列（font series）
   這是指字型的 weight（胖瘦）及 width（長扁）來區分的。例如粗、細字體，一般用的是 medium，粗體則是 bold。表3.2是一些例子：

Table 3.2: 一些字型系列

| 簡稱 | 代表意義 |
|---|---|
| m | medium |
| b | bold |
| bx | Bold extended |
| sb | Semi-bold |
| c | Condensed |

4. 字形（font shape）
   就是字的形狀。例如意大利斜體（italic）、斜體（slant）、small caps 等等。表3.3是幾個例子：

Table 3.3: 一些字形

| 簡稱 | 代表意義 |
|---|---|
| n | 正常字（normal），指 upright 或 roman |
| it | Italic |
| sl | Slanted |
| sc | Small Caps |

5. 字型大小（font size）
   預設的字型大小是 10pt（10 point）。不加單位的話，預設的就是 pt。

請注意，非標準 LaTeX 類別的預設字型大小可能會不一樣。

我們對字型要調整改變的，就是這些字型屬性的設定值。LaTeX 已設定好方便的指令給我們使用。

### 3.7.2 調整字族、字型系列、字形的指令

Table 3.4: 調整文字指令

| | 字型 | 標準指令 | 宣告式指令 (環境) | 舊用法 |
|---|---|---|---|---|
| 字形 | textup | \textup{textup} | {\upshape textup} | |
| | *italic* | \textit{italic} | {\itshape italic} | {\it italic} |
| | *slant* | \textsl{slant} | {\slshape slant} | {\sl slant} |
| | small caps | \textsc{small caps} | {\scshape small caps} | {\sc small caps} |
| 系列 | medium | \textmd{medium} | {\mdseries medium} | |
| | **boldface** | \textbf{boldface} | {\bfseries boldface} | {\bf boldface} |
| 字族 | roman | \textrm{roman} | {\rmfamily roman} | {\rm roman} |
| | sans serif | \textsf{sans serif} | {\sffamily sans serif} | {\sf sans serif} |
| | typewriter | \texttt{typewriter} | {\ttfamily typewriter} | {\tt typewriter} |

先別嚇了一跳，這是有跡可循的。其中 upright, medium, roman 都是一樣的，這是一般的正常字，就不必麻煩去設定他了，除非是要在特定字型範圍裡頭，重新改變成正常字體。從前面所說的簡稱的字串，再和 text, family, series, shape 去配對來使用，這樣只要記得簡稱就行了，例如：italic 的就是 \textit{}。不然也可以使用舊用法。但如果使用舊用法，那有時組合式的表示時可能會無效，例如粗斜體這種粗體和斜體設定混合時，就無法產生粗斜體了，這時還是得使用正統標準 LaTeX 的表示法。

要注意的是，大括號的位置，宣告式的指令，整個作用範圍是連指令一起包住的，他可以當成環境來使用，例如 \begin{itsahpe}, \end{itshape}，這樣在這個環境內的文字就通通會使用 italic 斜體，也可以不加參數使用，例如 \itshape，這樣以下的文字通通會使用 italic 斜體，直至另一個改變字型的指令出現為止。標準指令的作用範圍則是當做指令的一個參數，這些參數是出現在指令後的大括號內的。現在就來實際編譯個例子試看看：

```
% example8.tex
\documentclass{report}
\title{\bfseries It's a troll level}
```

```
\author{DGR\thanks{Thanks to the creator.}
        \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
\chapter{Checkpoint city!}
\section{It's \textsl{a} beautiful \textsl{city}}

\itshape
Progress city!
\upshape

\textbf{\textsl{Right}}?

\section{that {\it checkpoint}\/ city!}

You \textbf{\textit{can}} all

``have'' some

\chapter{second \textsc{checkpoint}}

It's \textsc{Christmas}

burny \textsc{stuff}
\end{document}
```

請注意，宣告式或舊用法，大括號是把指令和文字整個括住的。由於章節標題原本就會轉換成粗體，所以章節標題的部份，粗體就不必重複設了。但這裡發現例子裡第二章標題中的 checkpoint 並沒有改變字體，而且以 latex 編譯時會產生以下的錯誤（這些訊息也會在檔名.log 中找到）：

```
...
LaTeX Font Warning: Font shape `OT1/cmr/bx/sc' undefined
(Font) using `OT1/cmr/bx/n' instead on input line 4.
...
LaTeX Font Warning: Some font shapes were not available,
defaults substituted.
...
```

現在我們看到了前面所談的屬性簡稱，這在 LaTeX 就會使用這種屬性來表示而發出訊息，這裡 OT1/cmr/bx/sc 就表示了 OT1 編碼，Computer Modern Roman 字族，Bold extended 系列，而且是 small caps 形狀的字型，錯誤訊息顯示，他並沒有定義，因此，這個字型將會使用預設的字型來代替，這裡就是以 n 正常形狀的 bx 系列字型來替代。所以，字型指令並不是都可以隨意組合的，有些是根本就沒有這種字型，有些則是沒有用巨集去定義好，這樣 LaTeX 就取不到字了，但別擔心，頂多就是使用預設的字型罷了！

另一個很奇怪的地方，就是第一章、第二節的標題，為什麼是 `{\it checkpoint}\/ city!}`？這個插入的 \/ 是什麼東西？這是 TeX 系統調整斜體字（包括 iatlic 及 slanted）和正常字之間的空白的一個指令，稱為 italic correction。這樣，在斜體字和正常字之間的空白才會正常。那為什麼其他的斜體指令沒有加這個調整呢？這是因為 LaTeX 巨集在設計時就有考慮到這個問題，所以 `\textit{}` 這類標準指令都會自動調整 italic correction，不必由我們手動調整。

另外，章節標題本就會自動轉換成粗體，標題上的 that ‘checkpoint’ city! 為什麼沒有變粗體？這在前面有提到過，這種舊用法有時是無法複合使用的，粗體又斜體的指令會用不上來。因此，建議盡量使用 LaTeX 的第一種標準指令來改變字型。

### 3.7.3 相對字型大小的調整

接下來談最後一個字型大小屬性的調整，這在使用上比較單純，只要知道指令就可以馬上拿來使用。但是 TeX/LaTeX 系統中，談到字型，裡頭一堆地雷，例如前面談到正常的內文字型大小是 10pt，現在如果想製作海報，

需要 64pt 的字的時候就會發現，設不出來了！正常 LaTeX 的定義，字型的大小範圍是在 5–24.88pt 之間，超出這個範圍的字需要其他的 package 的幫忙。

這裡我們先來看看內文 10pt 時各種字型大小指令、實際例子及其大小（這是相對大小，會隨內文預設字型大小而自動調整），如表3.5：

Table 3.5: 相對字型大小

| 指令 | 實際例子 | 效果 | 實際的大小 |
|---|---|---|---|
| `\tiny` | `{\tiny tiny}` | tiny | 5pt |
| `\scriptsize` | `{\scriptsize scriptsize}` | scriptsize | 7pt |
| `\footnotesize` | `{\footnotesize footnotesize}` | footnotesize | 8pt |
| `\small` | `{\small small}` | small | 9pt |
| `\normalsize` | `{\normalsize normalsize}` | normalsize | 10pt |
| `\large` | `{\large large}` | large | 12pt |
| `\Large` | `{\Large Large}` | Large | 14.4pt |
| `\LARGE` | `{\LARGE LARGE}` | LARGE | 17.28pt |
| `\huge` | `{\huge huge}` | huge | 20.74pt |
| `\Huge` | `{\Huge Huge}` | Huge | 24.88pt |

這些字型大小指令也可以當成環境來使用，例如：

```
\begin{small}
   本文內容
\end{small}
```

## 3.8 原文照列

一般 LaTeX 遇到倒斜線會認為是一個指令的開始，如果連整個指令都要印出的時候呢？這時就要用到原文照列的指令及環境了。

### 3.8.1 原文照列指令

如果只是一小段的文字要原文照列，那使用指令會比較方便，這個指令就是 \verb|文字內容|，其中的 | 這個符號可以使用其他非字母的符號代替，只要前後相同就行了，例如：\verb+文字內容+ 這樣也是可以的。

### 3.8.2 原文照列環境

如果是一整段的內容要原文照列的話，使用環境會比較方便，那便是 verbatim 環境。不管是哪一種原文照列的情形，預設是使用打字機字族的字型來顯示的。底下是一個簡單的例子，說明原文照列指令及環境的使用：

```
% example9.tex
\documentclass{article}
\begin{document}
The example of \verb|\verb{}| command and
\texttt{verbatim} environment.

\section{\textbackslash{}\texttt{verb} command}

When you want to express you home directory, you can
 \verb|echo $HOME| varient to display your home directory
in your sh script.

\noindent
\verb*|This is    4 space here.|

\section{\texttt{verbatim} environment}

Here is a sh script to determine if on GNU/Linux system.

\begin{verbatim}
#!/bin/sh
```

```
Date=`date '+%y%m%d'`
if [ `uname` = Linux ]
then
  Mail=/var/spool/mail/edt1023
  Target=/mnt/hd
else
  Mail=/var/mail/edt1023
  Target=/mnt/pub
fi
\end{verbatim}
\end{document}
```

這裡會發現一些奇怪現象，例如 \verb* 那個星號就是讓空白以␣的方式表示出來的意思，verbatim 環境也是可以這樣使用。例如：

```
\verb*|This is    4 space here.|
```

也可以寫成:

```
\begin{verbatim*}
This is    4 space here.
\end{verbatim*}
```

差別在於，環境的上下行會多空出個空白行出來。

另外，標題為什麼不使用 \verb|\verb| 就好了呢？原因是原文照列的指令和環境都不能當做其他指令的參數，標題本身就是一個指令，所以 \verb 不能在裡頭。

使用 \textbackslash 這麼長的敘述，而不用 $\backslash$ 這個簡單的方式，原因是這個文稿有使用 LaTeX2HTML 來轉成 HTML 格式，使用 LaTeX 的替代表示法會轉成一般的符號，但使用後者的方式則會轉成圖檔，所以這裡就使用 LaTeX 的替代表示法了。

## 3.9 加入中文

這裡只說明如何使用 xeCJK 的情形，此時要改用 XeLaTeX 編譯，匯入套件就可以使用中文，底下由例子來說明。

```
\documentclass{article}
\usepackage{fontspec} % 設定字體為標楷體使用
\usepackage{xeCJK} % 使用 xsCJK 巨集套件
\setmainfont[Mapping=tex-text]{Times New Roman} % 設定英文字體
%\setCJKmainfont{cwTeXKai} % 設定中文字體，如果錯誤就改用作業系統的已安裝字
\begin{document}
\section{CJK測試}
顧名思義
\section{在等指令}
大家同心作戰
讓匈奴絕望
\end{document}
```

# Chapter 4

# 空間與位置

## 4.1 版面大小

我們對於所能控制的一整張紙的範圍都可以稱為版面。當然，我們的內文（body）並不是佔滿整張紙的範圍，上下左右都會留有一定的空白。當然，在內文以外的空白，也並非全是空白，他包含了頁足（footer），頁眉（header）及邊註（marginal note）的部份，記載關於頁數、註解等資訊。

### 4.1.1 紙張大小

表4.1列舉可宣告的紙張大小:

Table 4.1: 紙張大小

| 紙張 | 大小 | 紙張 | 大小 |
|---|---|---|---|
| a4paper | 21x29.7cm | letterpaper | 8.5x11in |
| a5paper | a5paper | legalpaper | 8.5x14in |
| b5paper | 17.6x25cm | executivepaper | 7.25x10.5in |

至於如何指定紙張大小，這裡先簡單說明一下這篇文章的設定，談到LaTeX 的文稿類別時 (第5章) 會再詳細說明。

```
\documentclass[12pt,a4paper]{report}
```

所以，這篇文章使用的是 a4paper，內文字型的大小是 12pt。方括號的參數

是選項，可以省略，如果省略的話，預設值就是 10pt/letterpaper。

## 4.2 調整橫向空間

### 4.2.1 調整橫向空間的環境

| | |
|---|---|
| \begin{center}...\end{center} | 讓這個環境內的內容置中 |
| \begin{flushleft}...\end{flushleft} | 讓這個環境內的內容靠左 |
| \begin{flushright}...\end{flushright} | 讓這個環境內的內容靠右 |

這可以方便的指定一個範圍的文句讓他作用，而不會影響環境以外的文句。其次，進入環境，縱使和上下行連在一起，沒有空出空白行，他也會自動的在上、下行空出個空白行出來，使用指令的話則不會。

## 4.3 條列環境

條列環境也是屬於一種空間的控制，他把一些文字按一定的方式來排列，條列環境中一些起頭的符號、文數字或字串，我們稱之為項目標籤（item label），利用這些不一樣的排列位置及不一樣的項目標籤起頭來敘述文句，就可以達到醒目的作用。這是以章節分隔以外，相當常用讓內容一目了然的方法，建議多多利用。請千萬記得，環境中還可以有環境，而且以下三種的條列方式可以混合交叉使用。

### 4.3.1 項目式條列環境（**itemize**）

這是以符號來起頭醒目的一種條列方式。例如：

```
\begin{itemize}
  \item 第一大項，這裡是第一大項。
  \item 第二大項，這裡是第二大項。
    \begin{itemize}
      \item 第一小項，這裡是第一小項。
      \item 第二小項，這裡是第二小項。
    \end{itemize}
```

```
    \item 第三大項，這裡是第三大項。
    \item 第四大項，這裡是第四大項。
\end{itemize}
```

排版出來會變成:

- 第一大項，這裡是第一大項。
- 第二大項，這裡是第二大項。
  - 第一小項，這裡是第一小項。
  - 第二小項，這裡是第二小項。
- 第三大項，這裡是第三大項。
- 第四大項，這裡是第四大項。

### 4.3.2 列舉式條列環境（**enumerate**）

這是以數目字或字母或羅馬數字來起頭醒目的條列方式。將上面例子 itemize 改成 enumerate 的話，會排版成：

1. 第一大項，這裡是第一大項。
2. 第二大項，這裡是第二大項。
   (a) 第一小項，這裡是第一小項。
   (b) 第二小項，這裡是第二小項。
3. 第三大項，這裡是第三大項。
4. 第四大項，這裡是第四大項。

### 4.3.3 敘述式條列環境（**description**）

這是以一個簡短文字敘述來起頭醒目的條列方式。改成 description 的話，他是以粗體文字來起頭醒目的，這些文字要用方括號括住。

```
\begin{description}
  \item[第一大項] 這裡是第一大項。
  \item[第二大項] 這裡是第二大項。
  \begin{description}
    \item[第一小項] 這裡是第一小項。
    \item[第二小項] 這裡是第二小項。
  \end{description}
  \item[第三大項] 這裡是第三大項。
  \item[第四大項] 這裡是第四大項。
\end{description}
```

排版出來的結果是：

**第一大項** 這裡是第一大項。

**第二大項** 這裡是第二大項。

**第一小項** 這裡是第一小項。

**第二小項** 這裡是第二小項。

**第三大項** 這裡是第三大項。

**第四大項** 這裡是第四大項。

要注意的是，不管哪一種的條列環境，每個項目（item）的文字敘述會自動折行，這相當方便，使用者只要把條列的結構弄妥，專心打每個項目的內容就成了。而且，如果使用方括號括住一些字元、字串或符號，那帶頭的標示將會是這些字元、字串或符號，如果是列舉式的條列方式，那麼有方括號的將不被編號，會自動跳過，編號順序則會自動順延。

# Chapter 5

# LaTeX 的標準文稿類別

這章主要是在說明 LaTeX 文稿的類別（document class），這是 LaTeX 規範文稿整體結構的方法。使用 class 的用意，就是把版面結構處理和實際文稿分開，這樣的最大好處就是維持整篇文章排版結構上的一致性，也使文稿內容更清爽簡潔，使用者只要專心於文稿內容的寫作即可，如果 class 定義的好，也可以達到一文多變化又不變更文稿內容的目的，只要把引用的 class 換成別的就可以了，其他的可以不必更動。

目前，LaTeX 標準類別用於一般文件，可用於一般的書信、雜誌、期刊、報告及論文。但有些期刊、論文會要求一定的結構，這時得依需求另行訂定。因此，也有其他的類別存在，標準類別並不是唯一的。甚至，也可以自行撰寫自己的文稿類別。當然，我們一般使用是不需要這麼講究，這裡只介紹 LaTeX 的標準類別。而且，如果有和他人交換文稿的需求時，我們應該盡可能的使用流通性較廣泛的類別。

## 5.1 LaTeX 類別的宣告

LaTeX 的類別，要在文稿的一開頭時就宣告（當然，其上有註解是沒有關係的），他的一般格式如下：

```
\documentclass[選擇性參數]{類別}
```

選擇性參數是可以省略的，但類別名稱則不能省，一定要指定一個類別。而且只能只有一個類別。

## 5.2 類別的選擇性參數

選擇性參數可以選擇多個，各個選項是以逗點分開的。

1. 10pt, 11pt, 12pt
   指定內文一般正常字的大小，預設是 10pt。其他點數沒有外來 package 的幫忙不能指定。

2. a4paper, letterpaper, b5paper, executivepaper, legalpaper
   指定紙張大小，預設是 letterpaper。

3. titlepage, notitlepage
   決定 title page 是否獨佔一頁。預設 article 不獨佔一頁，但 report/book 則會獨佔一頁，在這裡是可以指定來變更預設行為，例如 article 文稿，指定 titlepage 的話，那 title page 就會獨佔一頁。

4. onecolumn, twocolumn
   文章單欄或兩欄式，預設是單欄，也就是不分欄。

5. twoside, oneside
   是否區分奇偶數頁。預設 article/report 不區分，book 則會區分。一般的書籍，在裝訂的部份，他的中央線稱為書脊，偶數頁會在打開書籍時的左方，而且其中內容會偏向書脊的中央（此時是向右），反之，奇數頁會在右，內容一樣會偏左向書脊，在 oneside 的情形則不做這樣的區分，不管奇偶頁都會在紙張的中央部位。

6. landscape
   橫向列印或縱向列印，預設縱向（portait）。

7. draft
   草稿式編譯，這時圖檔將不會被引入，可加快編譯的速度。不過，如果編譯是使用向量字型的話，編譯速度應該是還算很快。但使用 draft 的一個好處是，過長的地方會標示出來。

8. openright, openany
   這是在控制，章的開始是否是奇數頁（right-hand page）。在 book 類

別，預設章會從奇數頁開始，report 類別則不會。article 類別沒有章，所以，對此一設定會忽略。

## 5.3 類別的種類

表5.1列出一般文章使用的類別，其他特殊巨集或 LaTeX 正式文件所使用的類別就不列出了，一般使用，這些類別就足夠了。

| 類別 | 一般用途 | 特性 |
|---|---|---|
| article | 一般短文 | 無章，連續頁方式的安排，無奇偶數頁的區分 |
| report | 較長論文 | 章會起新頁，預設無奇偶數頁的區分 |
| book | 書籍類 | 章會於奇數頁起新頁，預設有偶數頁的區分 |
| letter | 信件 | 英文信件格式 |

Table 5.1: 類別的種類

當然，這些用途並不是固定不變的，得看使用者的安排，不想多花時間、精神的話，那就依 LaTeX 預設的格式去使用，至少就不會太離譜。

# Chapter 6

# 表格的處理

這是屬於一般人覺得比較困難，但卻是很重要的部份。LaTeX 的表格，因為是抽象邏輯的思考方式來製作表格，對一般使用者而言，比較不容易轉換成直觀印象。當然，有些編輯器，例如 GNU Emacs，有方便畫 LaTeX 表格的編輯器 script，但這些我們先不去理他，先從 LaTeX 本身表格的結構理解起，這樣在使用其他的輔助工具時也會比較得心應手，甚至沒有其他工具，只要把握住表格的大結構，製作表格就不會摸不著頭緒了。

## 6.1 表格的種類

表格的使用，在文章上常常是必備的要件，他有歸納及醒目的作用，當然，表格太多也是會喧賓奪主。通常，我們中文的使用習慣，表格就是大方框內有小方框，文字置於小方框內，甚至某些小方框內還有斜線在分隔。為了排版上的方便及視覺表現上的美觀、清楚，在國際上大部份較正式的論文已不使用縱線、斜線，表格通常由橫線來做區隔，甚至完全沒有線條，使用空間區隔的方式。這種趨勢幾乎在二十幾年前就已開始普遍，只是國內的文件似乎還是很喜歡有縱、斜線在表格之中，好像沒有一些框線層層包住就不像表格。如非特殊的表現上的需求，我們應該朝簡化表格本身的方向走，將重點置於表格的內容及表格的邏輯結構安排。

另外，等粗的雙線條，可能也是得盡量避免，通常粗細不等的外框雙線條有裝飾的作用，因此，如果文件是較正式的論文，那就可能要避免，如

果是海報、DM 或要讓人們填寫的表格之類的，那又是另外一回事，這時封閉性的方框可能會有需要。這些規範只不過是一些慣例，並非一成不變的，得視文件的性質及使用場合來做變化，一個大原則是，如果是以文字敘述為主的文件，那麼，表格本身如果比文字內容搶眼太多的話，或許就要考慮簡化表格本身了。

## 6.2 tabbing 環境

這是 LaTeX 裡頭最基本的表格形式，除非自行另外定義、繪製，他並沒有方便可用的線條指令來區隔，完全使用空間、位置的配置來顯示表格內容，這時整個 tabbing 表格在 LaTeX 的地位並不是一個最小單位的 box，LaTeX 不會把整個表格當成一個單位來處理。所以，tabbing 表格是可以跨頁的，他可以被分成兩半來處理。因此，要和其他文字、圖表並排排版時，得另外放進一個 box 中，讓他自成一個 box 單位，例如 \parbox 或 minipage 環境裡頭

在 tabbing 環境中，第一個列（row）是以 \= 來標示 Tab 寬度來區隔欄位（column），這個寬度是由欄位裡頭的字串寬度所決定的。後續的每個欄位是由 \> 這個符號來區隔，每列尾要自行加上 \\ 來換行，最後一行可以不必使用 \\ 換行。tabbing 的基本大結構是：

```
\begin{tabbing}
  column1 \= column2 \= column3 \\
  item1   \> item2   \> item3   \\
  itemA   \> itemB   \> itemC
\end{tabbing}
```

這裡特意把他排列整齊（事實上，不排整齊 LaTeX 也會幫忙排好），這樣才能看得出來他的表格結構。那如果想調整欄位寬度時可以使用 template 的方式，例如：

```
\begin{tabbing}
  xxxxxxxxxx\=xxxxxxxxxx\=xxxxxxxxxx \kill
```

```
  column1 \> column2 \> column3 \\
  item1   \> item2   \> item3   \\
  itemA   \> itemB   \> itemC
\end{tabbing}
```

這裡以 10 個 x 為欄位的寬度，這裡的 \kill 表示這一行是不印出來的，只是在表示各個欄位的樣本寬度，而且他會自動換行。當然，要使用其他的字串也是可以，例如以表格中最長字串來取代整個 x 字串，這樣就會讓欄位寬度剛好都可以容納其他欄內內容。也可以使用 \hspace{6em} 或其他的長度指令，來指定欄位的寬度。

對於欄位內文字的控制，tabbing 較不完備，雖然 LaTeX 有提供 \' 讓這個符號之前的文字靠左，及 \` 讓這個符號之後的文字靠右，但實際運用，可能不是使用者想要的結果，因此 LaTeX 的表格，主要還是以 tabular 環境較為常用。但 tabbing 環境的好處是，他不見得一定要用於表格的排版，例如他也可以表現如條列環境般的另一種表現方式，而且他可以跨頁排版。

## 6.3 tabular 環境

這大概是最常使用的表格形式，可以很方便的畫線框。這種表格，LaTeX 是把整個表格當成一個單位來處理，就像字母一樣，因此他在版面的安排上是和一般的字母一般的處理，所以，這種表格不經特殊處理，無法被分割成兩個部份來跨頁。

和 tabbing 環境的不同，除了可以有線條之外（tabular 環境，當然也是可以完全沒有線條），分隔欄位的符號是 &，而且，一定要指定欄內文字的置放位置，欄內文字超出指定的寬度時，會自動折行，還有許多其他更細節的調整。

### 6.3.1 tabular 表格的基本結構

```
\begin{tabular}[t]{lll}
  \hline
```

```
    column1 & column2 & column3 \\
    \hline
    item1   & item2   & item3 \\
    itemA   & itemB   & itemC \\
    \hline
\end{tabular}
```

其中 [t] 表示 top，也可以是 b 表示 bottom，或 c 代表 center，這要在前後有文字相並排的時候才會顯現作用，因為 LaTeX 會把整個 tabular 表格當成一個字母單位，所以可以和其他文字、圖表並排排版。這些參數的意思是和同行文字的對齊方式，top 是表格頂端和前後文字對齊，bottom 則是表格底部和前後文字對齊，center 則是和表格中央對齊。

換行的方式和 tabbing 環境一樣，其中的 \hline 是畫一條橫線的意思，連續兩個 \hline\hline 會畫雙橫線，他本身會自動換行，因此不必加上換行符號。其中 \begin{tabular}{lll} 的 lll 是在指定各欄位內容在小方框內的置放位置，l 表示靠左（left），r 表示靠右（right），c 表示置中（center）。在 {lll} 中加上 bar（|）會畫縱線，例如 {|l|l|l|} 這樣就會變成傳統的大方框、小方框的表格。而兩個 bar 就會畫雙縱線。

tabular 環境內尚可使用另一個 tabular 環境來製作更複雜的表格，這在 tabbing 環境是不被允許的。

### 6.3.2 tabular 環境對欄位的調整

1. p{寬度}
   這裡的 p 指的是段落（paragraph）。通常用於一個小段落的文字，指定了寬度後裡頭的文字會自動折行，而且這個段落的頂端會和其他欄位的頂端對齊。

2. @{文字、符號或指令}
   這可以作用在本欄的各個列，讓他們都出現某個文字、符號或都在某個指令的作用下。這個指令另外會同時將欄位間距縮成 0，置於首尾

的話，會有讓橫線和文字切齊的作用（預設不會切齊，橫線兩端會多出欄位間距的部份）。

3. \multicolumn{欄位數}{左右位置}{文字內容}
   跨欄排版，例如一小段文字跨兩欄。左右位置可使用 lrc 之一。

4. \cline{a-b}
   畫某部份欄位的橫線，其中的 a-b 指的就是要畫線的欄位數，例如 \cline{2-3} 就是畫第二欄至第三欄的橫線。

5. \arrayrulewidth=單位長度
   調整表格線條的粗細，預設值是 0.4pt。使用方法：\arrayrulewidth=1.5pt 即可，但要注意的是要在進入 tabular 環境之前設定好。

6. \tabcolsep=單位長度
   調整兩欄位的左右間距。請注意，這個值是實際兩欄位間距值的一半，預設是 6pt。使用方法和 \arrayrulewidth 一樣。

7. \doublerulesep=單位長度
   調整畫雙線時，這兩線間的間距，預設值是 2pt。使用方法和 \arrayrulewidth 一樣。

8. \arraystretch
   調整表格的上下行距。請注意，這要由 \renewcommand 來重設，因為在 LaTeX 定義出的一個常數值，而這個 \arraystretch 只是這些常數值的倍數，我們要重新改變他才能改變預設倍數。

在 tabular 環境的參數中，可能是取代原來的參數，例如 p{}。也可能是置放在原參數的前後，如 @{}，這看一下實際例子就可以瞭解：

```
% example16.tex
\documentclass{article}
\usepackage{textcomp}              % for \textcelsius
\renewcommand{\arraystretch}{1.2} % 將表格行間距加大為原來的 1.2 倍
\arrayrulewidth=1pt                % 調整線條粗細為 1pt
```

```
\tabcolsep=12pt                    % 調整欄間距為 24pt
\begin{document}
\centering
\section*{SPECIFIC HEATS (20 \textcelsius\ AND 1 ATM)}
  \begin{tabular}{@{\sf }lll@{}}    % 第一欄位用 sans serif 字
    \hline
     & \multicolumn{2}{c}{\bf Specific Heats} \\ % 跨二三欄排版，文字置中
    \cline{2-3}                      % 只畫二三欄橫線
     & $c$ (J/kg$\cdot$K) & $C$ (J/mol$\cdot$K) \\
    \hline
    Aluminum     & 900  & 24.3 \\
    Copper       & 385  & 24.4 \\
    Gold         & 130  & 25.6 \\
    Steel/Iron   & 450  & 25.0 \\
    Lead         & 130  & 26.8 \\
    Mercury      & 140  & 28.0 \\
    Water        & 4190 & 75.4 \\
    Ice ($-$10 \textcelsius) & 2100 & 38 \\
    \hline
  \end{tabular}
\end{document}
```

@{} 如果完全沒有加入任何參數，那麼他的作用只是在去掉左右兩欄間距而已，可以把有關 @{} 的部份拿掉，試著再編譯看看，仔細比較看有什麼不同。有些專業排版的專家建議把表格前後加個 @{} 去除突出來的橫線(實際上就是去除原有左右兩邊間距的部份)。如果 @{} 裡頭不是指令，而是文字或符號，那這個文字或符號會加在各欄文字內容的前或後。

p{} 指令的使用時機是某一個欄位的文字比較多，需限定欄位的寬度讓他自動折行的情形，例如以下的例子：

```
% example17.tex
\documentclass{article}
```

```
\renewcommand{\arraystretch}{1.2} % 表格行間距為 1.2 倍
\begin{document}
\centering
\section*{Yi Syllables Area Character Blocks}
  \begin{tabular}{@{}llp{6cm}@{}}
    \hline
    Start & End & Character Block Name \\
    \hline
    A000  & A48F  & Yi Syllables.

                    Yi also known as Lolo, is a script
                    resembling Chinese in overall shaps
                    that is used in the Yunnan province
                    China. \\
    A490  & A4CF  & Yi Radicals.

                    Basic units of the Yi syllables. \\
    \hline
  \end{tabular}
\end{document}
```

這樣會把 p 指定的欄位當成一整個段落來處理，空一個空白行，同樣是表示新段落的開始。

## 6.4 浮動環境

tabular 表格，LaTeX 都會把他視為一個字母單位在處理，他不能被分割，常常因為圖表稍大些 LaTeX 就會起新頁去置放，但這樣一來原本的頁面就會顯得空盪，整個版面看起來很不自然，這種情形下，置放位置就很重要了，使用浮動環境的話，LaTeX 會繼續文字的部份，而把圖表置放在下一頁的頂端。通常在 LaTeX 的浮動環境下，圖表通常會置放在一頁的頂端或

都是底部，正常是不置放在一頁中間的位置，除非強迫指定，有放不下的情形時，就會讓他佔一整頁。因此，LaTeX 就得把前後位置經過整體的計算後再來決定圖表應該置放在什麼地方，這就是所謂的浮動環境。

### 6.4.1 基本的浮動環境

LaTeX 的浮動環境就是把表格置於 table 環境當中。在裡頭有 \{caption} 指令可以指定表格的標頭，而且編譯後會自動標上 'Table n:' 字樣，後接 caption 的內容，那個 n 會自動編號。一般國際上較正式的文件，caption 置放的位置慣例是表格的標題是置於表格上方，圖形則在下方。

```
  ...
\begin{table}[置放位置選項] % 進入浮動環境
  \caption{表格的標題}
  \begin{tabular}{表格參數}
    表格內容
  \end{tabular}
\end{table}
```

### 6.4.2 浮動環境選項參數

LaTeX 的浮動環境的配置，有時會不符和我們實際上的期望，這時可加入選項參數，如表6.1。如果都沒有指定，那預設是 [tbp]。

Table 6.1: 置放位置選項

| 位置選項 | 作用 |
|---|---|
| h | 置於下指令處位置 (here) |
| t | 置於一頁的頂端 (top) |
| b | 置於本頁底部，如空間不夠會置於次頁 (bottom) |
| p | 單獨佔一頁，此頁沒有內文的部份 (page) |

## 6.5 圖形的引入

這裡我們使用 graphicx package 來說明，他會自動引入 graphics package，這兩個 package，主要是一些指令的參數用法不同，由於 graphicx 的參數用法彈性較大，而且也和 LaTeX 的一些參數的形式較符合，因此，我們就以 graphicx 來說明，引入巨集時就引用 graphicx 就可以了。

### 6.5.1 引入外來圖檔的方法

使用 graphicx package 的 \includegraphics 指令，不使用浮動環境也是可以的。通常目前的 graphicx 巨集會自動判斷圖檔格式，所以延伸檔名可以不必寫上:

```
  ...
\usepackage{graphicx}
  ...
\begin{document}
  ...
\begin{figure}[置放位置選項]          % 進入浮動環境
  \includegraphics[選項參數]{圖檔名稱}
    ...
  \caption{圖的標題，可拿掉} % 標題在圖下，圖號會自動編號
  \label{引用圖檔的ID，可拿掉}   % 一定要在 caption 之後
\end{figure}
  ...
```

置放位址選項與 table 選項相同，如表6.1

### 6.5.2 includegraphics 指令的選項參數

這些選項參數可以有多個，各選項間以逗點來分隔，他的值的設定是使用等號（請注意，我們這裡談的是 graphicx 巨集，而不是 graphics，這在參數使用上不同）。

1. bb

   設定圖檔的邊界（bounding box），含四個值，每個值以空白隔開。例如 bb=98 98 468 430，這個意思就是左下角的座標是 (98, 98)，而右上角座標是 (468, 430)，這個參考標準是可被印出紙張的左下角為 (0, 0)。請注意，如果沒有指定單位的話，那預設是 bp。而且，這個設定在 pdflatex 會不被接受，此時請改使用 trim 選項參數。

   通常，這是會加上 clip 參數，作用是在修剪引入的圖檔的四周，但不是很好控制，所以建議圖檔由圖形處理或轉換程式去處理過後再引入會比較好控制。不加 clip 參數，加個星號 `\includegraphics*` 作用是一樣的。

2. clip

   修剪圖的四周指定的邊緣。

3. trim 作用和 bb 一樣，也是四個參數，但這裡指的是要去除的部份長度值，而非相對於左下角的相對座標。這個參數可以用在 pdflatex。例如：

   ```
   \includegraphics[trim=7 7 7 7, clip]{some}
   ```

   這會除去 some 這個圖檔的四周 7bp 的寬度。請注意，圖檔盡量不要加延伸檔名，讓系統自己去判斷，這樣文稿會比較有彈性。

4. angle

   旋轉的角度。旋轉指的是逆時針的方向轉的，除非使用負數的角度。

5. origin

   旋轉的中心點。

6. width

   這是指圖形的寬度，會自動伸縮調整，長度亦會等比例調整，使用 `width=\textwidth` 可使圖片與文章同寬

7. height

   這是指圖形的高度，會自動伸縮調整，寬度亦會等比例調整。

8. totalheight

   這是指圖形的總高度，即 height 再加上 depth 的值。會自動伸縮調整，寬度亦會等比例調整。

9. scale

   按一定比例縮放，這沒有單位，這是縮放倍數。

### 6.5.3 指定圖檔的搜尋路徑

如果圖檔很多，一個比較方便的方法就是在目前工作目錄下，新開一個子目錄來專門置放圖檔，這樣在文件的維護上也會比較好維護。LaTeX 系統預設找圖檔的路徑是 TeX 預設會去找的路徑及目前的工作目錄（通常目前的工作目錄會先找）。

```
\graphicspath{{路徑一}{路徑二}...} % 相對或絕對路徑皆可
\graphicspath{{./images/}}  % 只有一個子目錄也不可省略大括號
\graphicspath{{:images:}} % Mac 系統的表示法
```

將此指令放在 preamble 區即可，也可以直接在 \includegraphics{} 的參數裡頭就直接把路徑寫進去，但這是最不建議的方法，不管效率或是文稿可攜性都會很差。

### 6.5.4 圖文的旋轉

我們常常會需要某些圖文在特別的情況下旋轉一下，來看看 \rotatebox 這個指令詳細的使用方法：

```
\rotatebox[選項參數]{角度}{圖文物件}
```

角度和 \includegraphics 的 angle 選項參數一樣，但使用方法則簡化了，直接寫上數值即可，當然預設是逆時針方向旋轉。選項參數的部份可以有三個小選項：

1. origin

   設定旋轉中心點的位置，可以使用 lrctbB 或其中兩個的組合，其中 B

代表的是基線（baseline），其他的依其英文字母就可理解他的意義，如 t 是 top，r 是 right，c 是 center。預設的位置是左下角，文字的話，則是左下角的參考點（reference point），旋轉就是以此點所構成的的軸心線來轉的。

2. x, y 這是以左下角為原點，直接設座標，來表示 origin 所能表現的更精確中心點位置。

3. units
   設定旋轉的特殊弧度。其中 $units = -360$，這樣會把預設的逆時針旋轉，變成順時針旋轉。

旋轉不限於簡單的圖文物件，甚至一整個表格、圖形環境都可以拿來轉。要注意的是，編譯成 *.dvi 檔的話，有可能 dvi viewer 會不支援旋轉效果的解讀，此時要把他由 dvips 來轉成 ps 檔，或直接使用 pdflatex 編譯成 pdf 檔，再來預視。

# Bibliography

[1] 大家來學 LaTeX
https://www.cs.pu.edu.tw/~wckuo/doc/latex123/node1.html

[2] overleaf
https://www.overleaf.com/learn