
LaTeX (tikz) 轉換為圖像

發佈 1.0.0

高斯羽 博士 (Dr. Gāo, Sī Yǔ)

2019 年 11 月 12 日

1	系統和軟件	3
1.1	轉換軟件的使用理由	3
2	軟件的安裝和配置	5
2.1	配置系統環境變量 <code>Path</code>	5
2.2	測試是否成功修改	6
3	極簡教程	7
3.1	轉換為 SVG	7
3.2	轉換為 PNG	7
3.3	轉換為 EMF	8
3.4	轉換為 EPS	9
4	LaTeX standalone 包的配置	11
4.1	standalone 的轉換命令配置	11
4.2	編譯命令	12
4.3	簡例	12
5	轉換流程	15
5.1	轉換為 SVG 之流程	15
5.2	轉換為 PNG 之流程	17
5.3	轉換為 EMF 之流程	18
5.4	轉換為 EPS 之流程	19
6	腳本詳解	23
6.1	mk_folder.bat 詳解	23
6.2	gs_split_pdf.bat 詳解	24
6.3	pdf_to_svg.bat 詳解	25
6.4	pdf_to_png.bat 詳解	25
7	一步到位	27
7.1	一步轉換成 SVG	27
7.2	一步轉換成 PNG	28
7.3	一步轉換成 EMF	28
7.4	一步轉換成 EPS	29
8	總結	31

此項目是一個關於把 LaTeX 文檔直接轉換為各種圖像的教程（在編譯 TEX 文件時，同時生成單獨的圖片）。此教程主要關注如何把 tikz 生成的，內嵌於 LaTeX 生成的 PDF 文件中的圖像轉換為各種格式的單獨圖片。

此項目會討論到的圖片格式如下

- SVG（矢量圖）
- PNG（位圖）
- EMF（Windows 系統上的矢量圖）
- EPS（印刷常用格式）

此項目的在於提供基於 Windows 系統的教程和例子。作者相信 Linux 用戶有能力獨自解決這個問題。

此教程會提供軟件安裝和配置指南，並會結合例子進行講解。

此教程認為用戶已經對 LaTeX 有一定的理解，因而不會對 LaTeX 中之各種進行詳解。

本教程將會詳盡講解流程。若只想快速使用而不在乎原理，可先閱讀[軟件的安裝和配置](#)然後按照[極簡教程](#)中之步驟執行即可。

此項目會用到如下的系統和軟件，請先保證你已安裝了它們。

- Windows 7 或 Windows 10
- `texlive` (2019 年的發佈，免費)
- LaTeX 的 `standalone` 包 (`texlive` 自帶)
- `pdftocairo` (`texlive` 自帶，用於 PNG 轉換)
- `pdf2svg` (Windows 編譯版，免費。用於 SVG 轉換)
- `inkscape` (0.92.4，免費。用於 EMF 和 EPS 的轉換)
- `ghostscript` (9.50，免費。用於 PDF 文件的分頁)

1.1 轉換軟件的使用理由

此項目選擇的轉換軟件主要基於以下理由。

- 此項目堅持所有使用的轉換軟件必須為免費
- 轉換結果必須是一頁 PDF 一張圖
- 當把 PDF 轉換為矢量圖時，必須為真正的矢量圖，而不是包裹在矢量格式中的位圖
 - 因此，此項目不使用 `ImageMagick`，因其在轉換矢量圖時經常柵格化
- 當轉為為矢量圖時，字體應該嵌入而不是柵格化
- 轉換命令應儘可能簡單
- 使用的軟件儘可能少以降低依賴性

備註： 儘管 `inkscape` 也可以進行 PNG 和 SVG 的轉換，但 `pdftocairo` 和 `pdf2svg` 自帶了多頁到單頁的功能，使用便利，而且安裝也簡易，故而用此二軟件分別進行 PNG 和 SVG 的轉換而不使用 `inkscape`。若使用 `inkscape`，則需要先調用 `ghostscript` 對 PDF 進行分頁，然後再轉換。

軟件的安裝和配置

軟件的下載鏈接可在[系統和軟件](#)獲得。若鏈接失效，請自行搜索。

- **texlive** 的安裝過程比較長，請耐心等待（根據網速而定，可能需要數十分鐘到數小時不等）。
 - **texlive** 自帶 **standalone** 包和 **pdftocairo**
- **inkscape** 和 **ghostscript** 可採用默認安裝或者改變一下安裝路徑（非 C 盤）
- 關於 **pdf2svg**，請在 [github](#) 頁面下載 zip 壓縮包，然後解壓出對應系統位元的版本（32 位或 64 位）。之後把解壓出來的文件夾路徑加到系統的環境變量 **Path** 中即可。
- 請保證所有軟件的路徑都加到系統環境變量 **Path** 中，否則 Windows 的 CMD 會無法找到它們（除非用完整路徑）。此點會在[配置系統環境變量 Path](#)詳述。

2.1 配置系統環境變量 Path

當在 CMD 中鍵入非完整路徑時，譬如調用 **pdftocairo** 時，只鍵入“**pdftocairo**”而不是它完整的安裝路徑時（譬如“**c:\一些文件夾\pdftocairo.exe**”），系統會查找保存在 **Path** 變量中的路徑，看能不能找到。故此，為了便利和兼容，一般情況下軟件安裝時都會把自身重要的路徑加到 **Path** 中。

當然，也有列外的情況，比如不用安裝的軟件（譬如 **pdf2svg**），或者用戶沒有修改環境變量的權限。這些情況下就需要手動把路徑加到 **Path** 中。權限不足的用戶需要管理員的幫助，或者進行提權。

在 Windows 上配置環境變量有好幾種方法，此處描述基於 Windows 7 的一種方法。Windows 10 的方法基本一樣，只不過微軟把界面做了一些優化。

警告： 注意，在改動環境變量時請先進行備份。

步驟如下。

1. 打開系統的控制面板
2. 點擊右上角的查看方式並設為大圖標
3. 點擊“系統”
4. 點擊“高級系統設置”
5. 點擊“環境變量”

6. 選中“系統環境變量”下的 Path
7. 點擊“編輯”
8. 在彈出的窗口中，複製所有路徑並保存到用以備份用的純文本文件
9. 在彈出窗口的路徑結尾，鍵入分號“;”（英文的），之後粘貼入需要加入的文件夾路徑（不要把文件的完整路徑加進去）（Windows 10 有友好的 GUI，不需要鍵入分號）
10. 點擊所有確認鍵

2.2 測試是否成功修改

假設加入的是 `pdftocairo` 的路徑，那麼，打開 Windows 的 CMD，鍵入如下命令：

```
pdftocairo --help
```

如果配置 Path 成功，那一系列的幫助信息將會顯示在 CMD 裡面。如果不成功，那 CMD 會說找不到 `pdftocairo`。

備註： 可能需要重啓電腦

你可能需要重啓電腦才能令環境變量生效。若重啓後仍沒有生效，則證明配置錯誤。

一般來說，`texlive` 會自動添加路徑，但 `inkscape`，`ghostscript` 和 `pdf2svg` 都需要手動添加路徑。

本章意在提供最簡短而必要的步驟，以使用戶快速上手。

在應用本章步驟前，請先保證所有需要的軟件和配置已完成。

3.1 轉換為 SVG

1. 把本教程附帶的 `util` 文件夾複製到需要轉換的 TEX 文件所在之目錄下。
2. 對需要轉換的 TEX 的文件的 `documentclass` 進行如下配置：

```
1 \documentclass[tikz, convert, convert={outtext=.svg, command=\unexpanded{
2 % 'out_svg' 是用 存放 SVG 的文件
3 % 'out_svg' 是用來存放 SVG 的文件夾
4 % 'out_svg' is the destination folder for SVG files
5 call ./util/mk_folder out_svg
6 && cd /d out_svg
7 && call ../util/pdf_to_svg ../\infile\space \outfile\space
8 }]{standalone}
```

3. 使用 `-shell-escape` 參數對 TEX 文件進行編譯。例如（需要把尖括號，及其所包裹的內容更換成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的
文件名>.tex
```

4. 轉換好的 SVG 文件將存放在 `out_svg` 文件夾下。

3.2 轉換為 PNG

1. 把本教程附帶的 `util` 文件夾複製到需要轉換的 TEX 文件所在之目錄下。
2. 對需要轉換的 TEX 的文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={command=\unexpanded{
2 % 'out_png' 是用 存放 PNG 的文件
3 % 'out_png' 是用來存放 PNG 的文件夾
4 % 'out_png' is the destination folder for PNG files
5 call ./util/mk_folder out_png
6 && cd /d out_png
7 && call ../util/pdf_to_png.bat 600 ../\infile\space
8 }]}{standalone}

```

3. 使用 `-shell-escape` 參數對 TEX 文件進行編譯。例如（需要把尖括號，及其所包裹的內容更換成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的
文件名>.tex
```

4. 轉換好的 PNG 文件將存放在 `out_png` 文件夾下。

3.3 轉換為 EMF

1. 把本教程附帶的 `util` 文件夾複製到需要轉換的 TEX 文件所在之目錄下。
2. 對需要轉換的 TEX 的文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={outext=.pdf, command=\unexpanded{
2 % 'out_emf' 是用 存放 EMF 的文件
3 % 'out_emf' 是用來存放 EMF 的文件夾
4 % 'out_emf' is the destination folder for EMF files
5 call ./util/mk_folder out_emf
6 && call ../util/gs_split_pdf.bat out_emf \outfile\space \infile\space
7 && cd /d out_emf
8 && call ../util/pdf_to_emf.bat
9 && del /F *.pdf \sapce
10 }]}{standalone}
11 % 有 的 PDF 的 PDF EMF，所以要先用 ghostscript 把 LaTeX 生
12 % 成的
13 % PDF 分 ，然后 用 inkscape 做循 ，把所有 的 PDF EMF，最后 除
14 % 所
15 % 有 的 PDF，只保留 EMF。
16 % inkscape 只能實現單張的 PDF 轉換 EMF，所以要先用 ghostscript 把 LaTeX 生
17 % 成的
18 % PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉換 EMF，最後刪除
19 % 所
20 % 有單頁的 PDF，只保留 EMF。
21 % inkscape can only convert single page PDF to EMF. Therefore, the whole
22 % PDF generated by LaTeX needs to be split into single pages first, by
23 % ghostscript. Then use inkscape in a loop to convert all single page
24 % PDFs into EMFs. Finally, delete all single page PDFs and keep only the
25 % EMFs.

```

3. 使用 `-shell-escape` 參數對 TEX 文件進行編譯。例如（需要把尖括號，及其所包裹的內容更換成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的
文件名>.tex
```

4. 轉換好的 EMF 文件將存放在 `out_emf` 文件夾下。

3.4 轉換為 EPS

1. 把本教程附帶的 util 文件夾複製到需要轉換的 TEX 文件所在之目錄下。
2. 對需要轉換的 TEX 的文件的 documentclass 進行如下配置：

```

1 \documentclass[tikz, convert, convert={outtext=.pdf,
2 command=\unexpanded{{
3 % 'out_eps' 是用 存放 EPS 的文件
4 % 'out_eps' 是用來存放 EPS 的文件夾
5 % 'out_eps' is the destination folder for EPS files
6 call ./util/mk_folder out_eps
7 && call ./util/gs_split_pdf.bat out_eps \outfile\space \infile\space
8 && cd /d out_eps
9 && call ../util/pdf_to_eps.bat
10 && del /F *.pdf \sapce
11 }}}}{{standalone}}
12 % inkscape 只能          的 PDF          EPS，所以要先用 ghostscript 把 LaTeX 生
13 成的
14 % PDF 分    ，然后 用 inkscape 做循    ，把所有    的 PDF          EPS，最后 除
15 所
16 % 有    的 PDF，只保留 EPS。
17 % inkscape 只能實現單張的 PDF 轉換 EPS，所以要先用 ghostscript 把 LaTeX 生
18 成的
19 % PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉換 EPS，最後刪除
20 所
21 % 有單頁的 PDF，只保留 EPS。
22 % inkscape can only convert single page PDF to EPS. Therefore, the whole
23 % PDF generated by LaTeX needs to be split into single pages first, by
24 % ghostscript. Then use inkscape in a loop to convert all single page
25 % PDFs into EPSs. Finally, delete all single page PDFs and keep only the
26 % EPSs.
```

3. 使用 -shell-escape 參數對 TEX 文件進行編譯。例如（需要把尖括號，及其所包裹的內容更換成你的 TEX 文件的文件名）：

```
xelatex -synctex=1 -interaction=nonstopmode -shell-escape < 你 TEX 文件的
文件名>.tex
```

4. 轉換好的 EPS 文件將存放在 out_eps 文件夾下。

本章中為了方便排錯，命令是分部進行的（通過 && 連成一行）。這些命令其實是可以放在同一個腳本中，而簡化 documentclass 的設置的。詳情請看[一步到位](#)。

LaTeX standalone 包的配置

本教程是基於由 [Martin Scharrer](#) 開發的包（自帶 `standalone` 類），故此於此對此包稍作講解。

備註: `standalone` (*complex*)¹

`standalone` 是 LaTeX 中非常有用的一個包。本教程主要講述怎樣利用此包來進行圖片的轉換，但此包其實還有其它相當多的應用。[Overleaf](#) 上有一個非常有用的教程。

4.1 standalone 的轉換命令配置

`standalone` 本身的 [說明文檔](#) 已經對配置有詳盡的說明，此處重點說一下轉換成圖片需要用到的 `convert` 選項。

配置 `convert` 需要在 `documentclass` 中進行。以下是一個利用 `pdf2svg` 轉換為 SVG 的範例配置。

```

1 \documentclass[tikz, convert, convert={outtext=.svg, command=\unexpanded{
2 pdf2svg \infile\space \outfile\space all
3 }]{standalone}

```

其中，

- `tikz` 此選項告訴 `standalone` LaTeX 文檔中存在 `tikz` 圖片。
- `convert` 此選項開啓 `standalone` 的轉換功能。
- `convert={}` 此選項是 `convert` 的詳細配置項。
- `outtext=.svg` 設置輸出文件的後綴名為 “.svg”。更詳細的說明請參看 `standalone` 本身的 [說明文檔](#) 中的表 1。
- `command=\unexpanded{}` 此項是將要調用系統運行的命令。
- `pdf2svg` 調用的轉換工具。

備註: `pdf2svg` 的語法

`pdf2svg` 的語法可以參看 [這裡](#)。

¹ Ghost In Shell : Standalone Complex

其中，將一多頁 PDF 轉換為分頁的多個 SVG 的語法為：

```
pdf2svg < 輸入文件名>.pdf < 輸出文件名>%d.svg all
```

注意，尖括號，及其所包裹中的內容需要替換為所需的文件名。

- `\infile` 輸入文件名，包含後綴名。默認後綴名為“.pdf”或“.ps”。更詳細的說明請參看 `standalone` 本身的 [說明文檔](#) 中的表 1。
- `\space` 空格。若不使用此參數，`\infile` 後不會有空格，無論你實際上鍵入了多少個。`\outfile` 也是這樣。
- `\outfile` 輸出文件名，包含後綴名。默認後綴名為“.png”。此處已經通過 `outext` 更改為“.svg”。更詳細的說明請參看 `standalone` 本身的 [說明文檔](#) 中的表 1。

SVG 配置範例中之命令將會被翻譯為如下（可以通過查看 LOG 文件確認）。其中，`mew_to_svg` 為所用的 TEX 文件的文件名。

```
1 pdf2svg mew_to_svg.pdf mew_to_svg-%01d.svg all
```

由此可以看出，轉換的重點，是要把 `convert={}` 中的配置正確設置，以令 LaTeX 將其翻譯成正確的系統命令來進行圖片的轉換。用戶可以把多個系統命令整合為一行，以做出豐富多彩的組合來達成不同的目標（在 Windows 中可以通過“&”或“&&”把多行命令合併為一行）。在 [轉換流程](#) 中將會詳細敘述各種圖片轉換的流程。

備註：運行系統命令

其實在本小結就可以看出，既然 `standalone` 可以調用以上的命令，那當然也可以調用其它系統命令。理論上，用戶可以調用各種命令來做各種事，不僅僅是圖片的轉換。如果你有興趣，應該可以做到編譯完後自動上傳到某個網絡位置，或者刪除整個硬盤這一類有趣的事情。

4.2 編譯命令

`standalone` 需要在編譯時使用 `-shell-escape` 參數。一個使用 `xelatex` 對 `mew_to_svg.tex` 進行編譯的命令如下（用 `xelatex` 是因為需要處理中文）。如果你使用 LaTeX 編輯器進行書寫，比如 `TEXstudio`，則需要在其中編輯其命令。你也當然可以直接在 TEX 文件所在之目錄下打開 CMD，用命令直接編譯。

```
1 xelatex -synctex=1 -interaction=nonstopmode -shell-escape mew_to_svg.tex
```

4.3 簡例

以下提供一個轉換為單頁多個 SVG 的簡例。詳細的例子會在後文說明。

以下的文件可以在此項目的根目錄和 `mew` 文件夾中找到。

主文件：

`mew` 文件夾中的 `mew_to_svg.tex`

```
1 % 是一 tikz 片 成多 SVG 的示例文件，使用 pdf2svg
2 % 這是一個將 tikz 圖片轉換成多張 SVG 的示例文件，使用 pdf2svg 來實現轉換
3 % This is a demo file for tikz to multiple SVGs using pdf2svg
4 \documentclass[tikz, convert, convert={outext=.svg, command=\unexpanded{
5 pdf2svg \infile\space \outfile\space all
```

(continues on next page)

(繼續上一頁)

```

6 }}}{standalone}
7
8 \usepackage{xeCJK}
9 \setCJKmainfont{Microsoft YaHei}
10
11 \usepackage{scalefont}
12 \usepackage{tikz}
13
14 % tikz 和 colour 的 定
15 % tikz 和 colour 的設定
16 % tikz and colour configs
17 \input{../configs_tikz.tex}
18 \input{../configs_colour.tex}
19
20 \begin{document}
21
22     % 全局字体 放
23     % 全局字體縮放
24     % global font scale
25     \scalefont{1.3}
26
27     % tikz 像文
28     % tikz 圖像文檔
29     % tikz pics file
30     \input{../tikz_pics.tex}
31
32 \end{document}

```

tikz 配置文件：

本教程根目錄中的 configs_tikz.tex

```

1 % 以下是 于 tikz 中 流程 的 置
2 % 以下是關於 tikz 中畫流程圖的設置
3 % configure flowchart shapes
4 \usetikzlibrary{shapes.geometric, arrows, positioning, calc}
5
6 % start, end shape
7 \tikzstyle{startstop} = [rectangle, rounded corners, minimum width=3cm,
8 minimum height=1cm, text centered, text=white, draw=black,
9 fill=colorStarstop]
10
11 % process shape
12 \tikzstyle{process} = [rectangle, minimum width=3cm, minimum height=1cm,
13 text centered, text=white, draw=black, fill=colorPro]
14
15 % decision shape
16 \tikzstyle{decision}=[diamond, minimum width=3cm, minimum height=1cm,
17 text centered, draw=black, fill=colorDec]
18
19 % comment shape
20 \tikzstyle{comment}=[dashed, draw=black, fill=gray!10, minimum width=3cm, minimum
21 ↪ height=1cm, text centered]
22
23 % docstring shape
24 \tikzstyle{docstring}=[draw=orange, fill=white, minimum width=50mm, text width=80mm,
25 ↪ minimum height=1cm]

```

(continues on next page)

```

24
25 % arrows shape
26 \tikzstyle{arrow} = [ultra thick,->,>=stealth, line width=1.5mm]
27
28 % comment shape
29 \tikzstyle{comment}=[dashed, draw=black, fill=gray!10, minimum width=3cm, minimum
↪height=1cm, text centered]
30
31 \tikzset{
32   subprocess/.style = {rectangle, draw=black, semithick, fill=orange!30,
33     minimum width=#1, minimum height=1cm, inner xsep=3mm, % <-- changed
34     text width = \pgfkeysvalueof{/pgf/minimum width}-2*\pgfkeysvalueof{/pgf/inner
↪xsep},
35     align=flush center,
36     path picture={\draw
37       ([xshift =2mm] \ppbb.north west) -- ([xshift= 2mm] \ppbb.south west)
38       ([xshift=-2mm] \ppbb.north east) -- ([xshift=-2mm] \ppbb.south east);
39     },
40   },
41   subprocess/.default = 24mm % <-- added
42 }% end of tikzset
43
44 \usetikzlibrary{positioning}

```

color 配置文件：

本教程根目錄中的 configs_colour.tex

```

1 % 以下是 色 的 置
2 % 以下是顏色的設置
3 % colour defs
4 \usepackage{color}
5 \definecolor{colorStarstop}{RGB}{174, 23, 21}
6 \definecolor{colorPro}{RGB}{0, 175, 121}
7 \definecolor{colorDec}{RGB}{255, 192, 0}
8 \definecolor{colorYes}{RGB}{51, 153, 51}
9 \definecolor{colorNo}{RGB}{255, 0, 0}

```

轉換流程

本章將對轉換流程進行講解。鑑於使用到之軟件，轉換成 PNG 的流程與轉換成 SVG 之流程相近，而轉換成 EPS 之流程則於轉換成 EMF 之流程同理。

本教程中轉換的流程將利用到數份 Windows 的批處理腳本（在本教程的 `util` 文件夾中），本教程將在[腳本詳解](#)中對它們進行詳細講解。

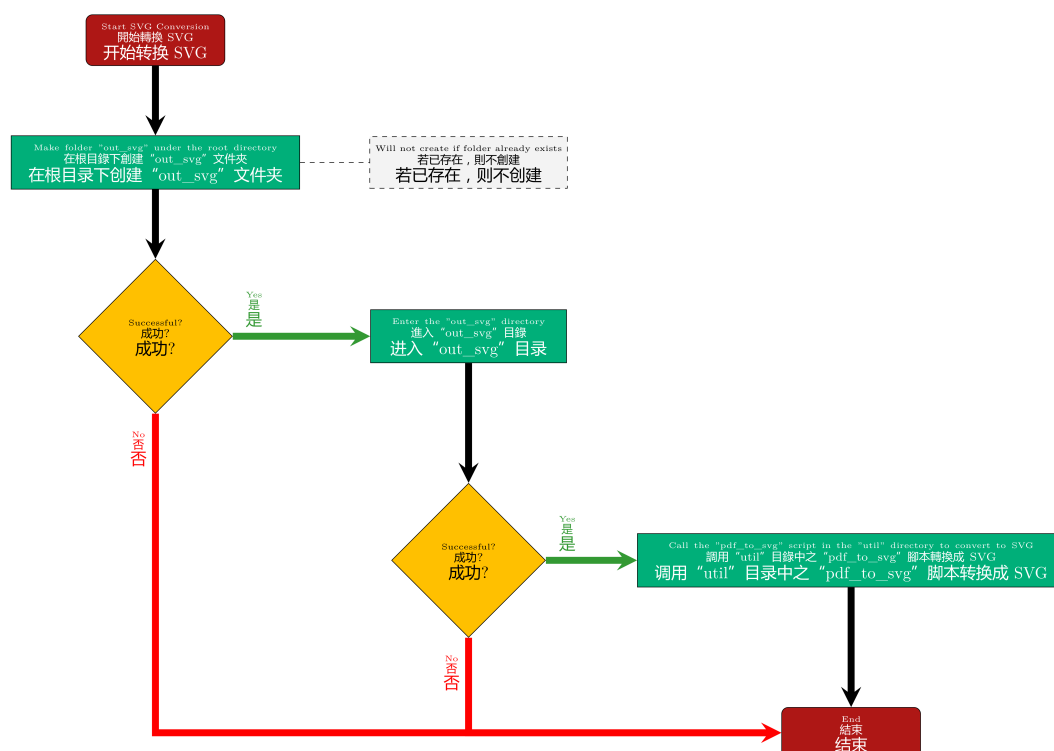
本章中之方法一律需要使用 `-shell-escape` 參數來進行編譯。一個使用 `xelatex` 對 `mew_to_svg.tex` 進行編譯的命令如下（用 `xelatex` 是因為需要處理中文）。

```
1 xelatex -synctex=1 -interaction=nonstopmode -shell-escape mew_to_svg.tex
```

請把 `mew_to_svg` 替換為你的 TEX 文件文件名。

5.1 轉換為 SVG 之流程

轉換為 SVG 之流程可用如下之流程圖表示：



如上圖所示，轉換流程將在所在之目錄創建一個名為 `out_svg` 的文件夾單獨存放生成的 SVG 文件。之後將會調用 `util` 文件夾中的 `pdf_to_svg` 腳本來實現轉換。

此流程需要用戶對需要轉換之 TEX 文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={outtext=.svg, command=\unexpanded{
2 % 'out_svg' 是用 存放 SVG 的文件
3 % 'out_svg' 是用來存放 SVG 的文件夾
4 % 'out_svg' is the destination folder for SVG files
5 call ./util/mk_folder out_svg
6 && cd /d out_svg
7 && call ../util/pdf_to_svg ../\infile\space \outfile\space
8 }]}{standalone}

```

備註：設置 `outtext`

當 `outtext` 有設置時，`standalone` 會自動地在輸出文件（即是 `outfile`）的文件名（不含後綴名）後加上計數關鍵字（一般是 `%d`）。

這小節的方法正是利用 `standalone` 之此特性，結合 `pdf2svg` 的語法來進行 PDF 轉換為分頁的 SVG。

警告：關於 “%” 和 “\” 符號

在 LaTeX 中，“%” 是一個保留字，用來表示註釋。如果直接使用在 `documentclass` 之中，則會把其後面的同行代碼全部註釋掉。這樣的話，編譯時會出問題。

然而，若用 “\” 進行轉義（即 “escape”）的話 `standalone` 是會把 “\” 符號作為明文加入到命令中的。這樣一來，命令通常都不對，因為 “\” 在 LaTeX 中表示的是後面跟的是參數或者命令。而在 Windows 命令中 “%” 通常用來指代參數，在 Windows 中使用 `for` 循環時絕對會用到它，無法避免。

綜上所述，如果在 `documentclass` 裡面直接把系統命令寫全的話，很難保證其正確性。

故此，作者選擇把命令封裝到多個批處理腳本中，這樣就可以避免以上提及的符號問題同時方便排錯。

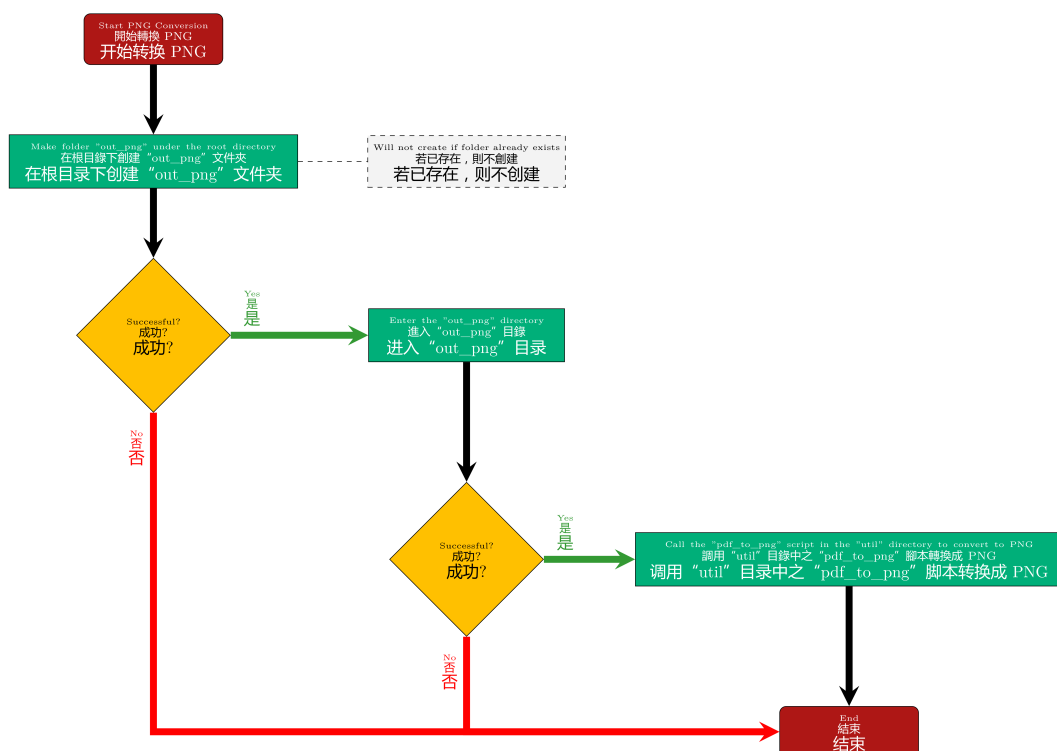
本方法用到以下兩份腳本：

1. `mk_folder` 創建文件夾。
2. `pdf_to_svg` 將多頁的 PDF 轉換為分頁的 SVG。

它們的詳細講解在腳本詳解 中。

5.2 轉換為 PNG 之流程

轉換為 PNG 之流程可用如下之流程圖表示：



如上圖所示，轉換流程將在所在之目錄創建一個名為 `out_png` 的文件夾單獨存放生成的 PNG 文件。之後將會調用 `util` 文件夾中的 `pdf_to_png` 腳本來實現轉換。

此流程需要用戶對需要轉換之 TEX 文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={command=\unexpanded{
2 % 'out_png' 是用 存放 PNG 的文件
3 % 'out_png' 是用來存放 PNG 的文件夾
4 % 'out_png' is the destination folder for PNG files
5 call ./util/mk_folder out_png
6 && cd /d out_png
7 && call ../util/pdf_to_png.bat 600 ../\infile\space
8 }]{standalone}

```

備註: `pdftocairo`

pdftocairo 會自動地把一多頁的 PDF 自動分割為多張 PNG。故此只需要把輸入文件給它即可 (即 infile)，而不需要設置輸出文件 (即 outfile)。

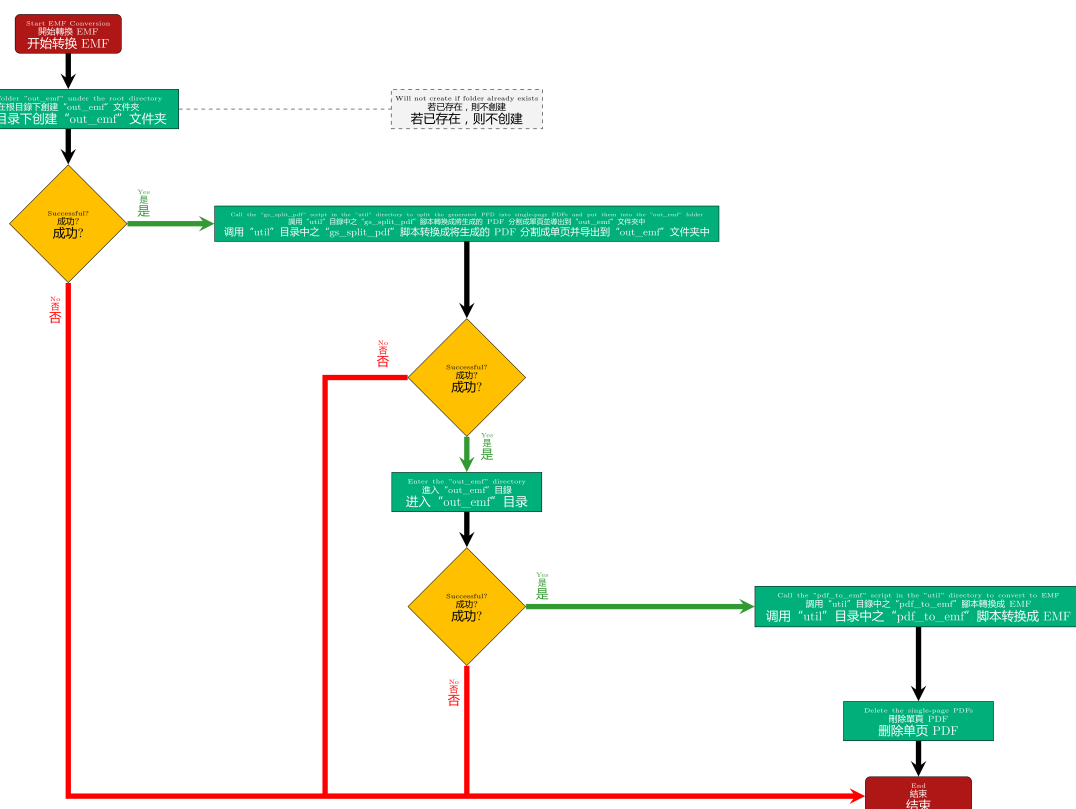
本方法用到以下兩份腳本：

1. **mk_folder** 創建文件夾。
2. **pdf_to_png** 將多頁的 PDF 轉換為分頁的 PNG。

它們的詳細講解在腳本詳解 中。

5.3 轉換為 EMF 之流程

轉換為 EMF 之流程可用如下之流程圖表示：



如上圖所示，轉換流程將在所在之目錄創建一個名為 `out_emf` 的文件夾單獨存放生成的 EMF 文件。之後將會調用 `util` 文件夾中的 `gs_split_pdf` 來對生成的 PDF 進行分頁，`pdf_to_emf` 腳本來實現轉換。轉換完畢後會刪除所有單頁之 PDF，只保留 EMF。

此流程需要用戶對需要轉換之 TEX 文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={outtext=.pdf, command=\unexpanded{
2 % 'out_emf' 是用 存放 EMF 的文件
3 % 'out_emf' 是用來存放 EMF 的文件夾
4 % 'out_emf' is the destination folder for EMF files
5 call ./util/mk_folder out_emf
6 && call ./util/gs_split_pdf.bat out_emf \outfile\space \infile\space
7 && cd /d out_emf
8 && call ../util/pdf_to_emf.bat
9 && del /F *.pdf \sapce
  
```

(continues on next page)

(繼續上一頁)

```

10 }}}{standalone}
11 % inkscape 只能      的 PDF      EMF, 所以要先用 ghostscript 把 LaTeX 生成的
12 % PDF 分      , 然後 用 inkscape 做循環 , 把所有      的 PDF      EMF, 最後 除所
13 % 有      的 PDF, 只保留 EMF。
14 % inkscape 只能實現單張的 PDF 轉換 EMF, 所以要先用 ghostscript 把 LaTeX 生成的
15 % PDF 分頁, 然後調用 inkscape 做循環, 把所有單頁的 PDF 轉換 EMF, 最後刪除所
16 % 有單頁的 PDF, 只保留 EMF。
17 % inkscape can only convert single page PDF to EMF. Therefore, the whole
18 % PDF generated by LaTeX needs to be split into single pages first, by
19 % ghostscript. Then use inkscape in a loop to convert all single page
20 % PDFs into EMFs. Finally, delete all single page PDFs and keep only the
21 % EMFs.

```

備註: inkscape

inkscape 在轉換時不會自動對 PDF 進行分頁。若直接使用 inkscape 對多頁 PDF 進行轉換, 只有第一頁會被轉換。

故此, 本方法會先調用 ghostscript 對 PDF 進行分頁, 然後在調用 inkscape 對所有的單頁 PDF 進行轉換。

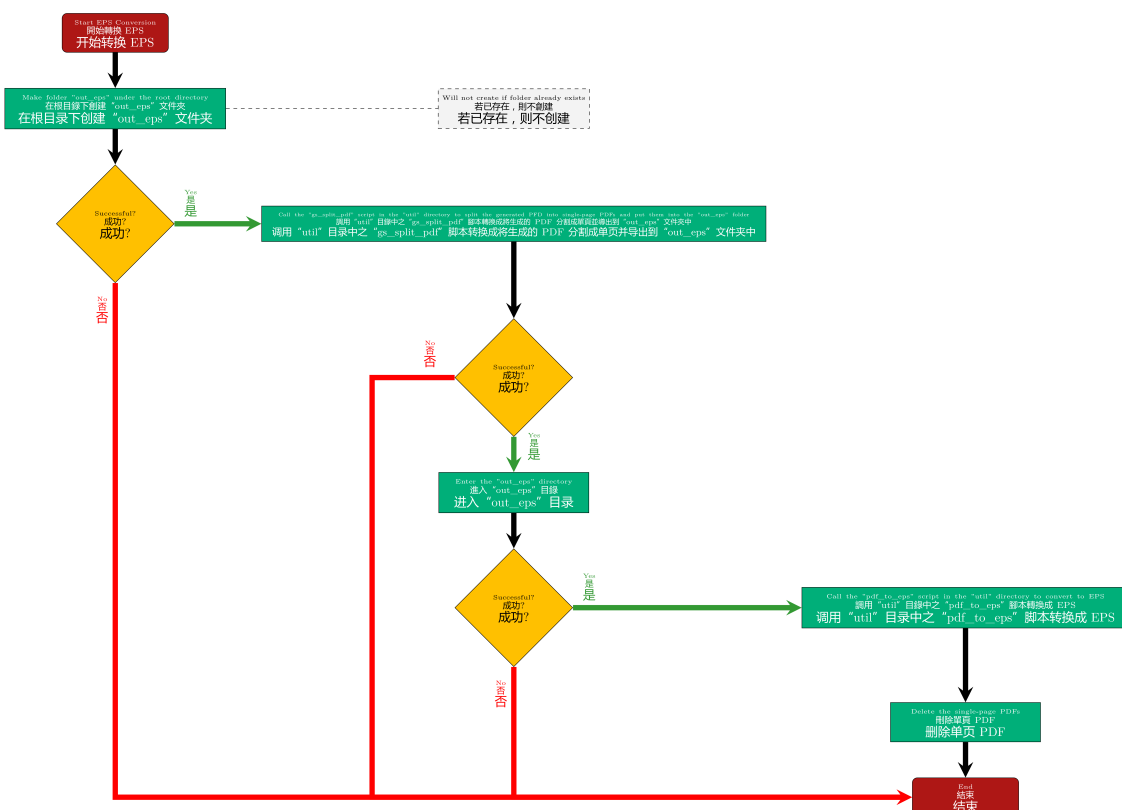
本方法用到以下三份腳本:

1. **mk_folder** 創建文件夾。
2. **gs_split_pdf** 將多頁的 PDF 分割為單頁的 PDF。
3. **pdf_to_emf** 將 PDF 轉換為 EMF (僅一頁)。

它們的詳細講解在腳本詳解 中。

5.4 轉換為 EPS 之流程

轉換為 EPS 之流程可用如下之流程圖表示:



如上圖所示，轉換流程將在所在之目錄創建一個名為 `out_eps` 的文件夾單獨存放生成的 EMF 文件。之後將會調用 `util` 文件夾中的 `gs_split_pdf` 來對生成的 PDF 進行分頁，`pdf_to_eps` 腳本來實現轉換。轉換完畢後會刪除所有單頁之 PDF，只保留 EMF。

此流程需要用戶對需要轉換之 TEX 文件的 `documentclass` 進行如下配置：

```

1 \documentclass[tikz, convert, convert={outtext=.pdf,
2 command=\unexpanded{{
3 % 'out_eps' 是用 存放 EPS 的文件
4 % 'out_eps' 是用來存放 EPS 的文件夾
5 % 'out_eps' is the destination folder for EPS files
6 call ./util/mk_folder out_eps
7 && call ./util/gs_split_pdf.bat out_eps \outfile\space \infile\space
8 && cd /d out_eps
9 && call ../util/pdf_to_eps.bat
10 && del /F *.pdf \sapce
11 }}}}standalone}
12 % inkscape 只能 的 PDF EPS，所以要先用 ghostscript 把 LaTeX 生成的
13 % PDF 分，然後用 inkscape 做循環，把所有 的 PDF EPS，最後 除所
14 % 有 的 PDF，只保留 EPS。
15 % inkscape 只能實現單張的 PDF 轉換 EPS，所以要先用 ghostscript 把 LaTeX 生成的
16 % PDF 分頁，然後調用 inkscape 做循環，把所有單頁的 PDF 轉換 EPS，最後刪除所
17 % 有單頁的 PDF，只保留 EPS。
18 % inkscape can only convert single page PDF to EPS. Therefore, the whole
19 % PDF generated by LaTeX needs to be split into single pages first, by
20 % ghostscript. Then use inkscape in a loop to convert all single page
21 % PDFs into EPSs. Finally, delete all single page PDFs and keep only the
22 % EPSs.
  
```

本方法之原理於轉換為 EMF 的流程完全一樣。都是先對生成的 PDF 進行分頁，再調用 `inkscape` 進行處理。

本方法用到以下三份腳本：

1. `mk_folder` 創建文件夾。
2. `gs_split_pdf` 將多頁的 PDF 分割為單頁的 PDF。
3. `pdf_to_eps` 將 PDF 轉換為 EPS（僅一頁）。

它們的詳細講解在[腳本詳解](#)中。

腳本詳解

本章將對本教程中所用到之腳本進行講解。

本項目所使用到的腳本存放在 `util` 文件夾之中，它們為：

- `mk_folder.bat` 用於創建文件夾。

備註：腳本命名問題

在 Windows 當中，如果把批處理腳本命名為所要調用的系統命令名，很可能會導致死循環。故此命名此腳本為 `mk_folder.bat` 而不是 `mkdir.bat`

- `gs_split_pdf.bat` 調用 `ghostscript` 把 PDF 分割為單頁。
- `pdf_to_svg.bat` 把多頁 PDF 轉換為分頁的 SVG。
- `pdf_to_png.bat` 把多頁 PDF 轉換為分頁的 PNG。
- `pdf_to_emf.bat` 把 PDF 轉換為 EMF（僅一頁）。
- `pdf_to_eps.bat` 把 PDF 轉換為的 EPS（僅一頁）。

警告：路徑

在設置 `documentclass` 時，需要注意輸入的路徑參數。本教程一律使用相對路徑。

`./` 是指當前所在路徑。

`../` 是指往上移一層目錄。

6.1 `mk_folder.bat` 詳解

`mk_folder.bat` 的內容如下：

```
1 @echo off
2 REM this file is not call "mkdir" to avoid run errors
3
```

(continues on next page)

(繼續上一頁)

```

4 set dst_dir=%1
5
6 if not exist %dst_dir%/NUL mkdir %dst_dir%

```

此腳本用於在 CMD 當前所在之目錄為根來創建文件夾。如果該文件夾已存在，則不創建而退出。

此腳本接受一個參數。此參數為將要創建的文件夾路徑。

此腳本之使用方法如下：

```
mk_folder < 要創建之文件夾路徑>
```

此腳本的語法和 `mkdir` 命令的語法一致。此處給出一個例子，假設要在目前的目錄下創建一個名為 `a` 的文件夾，其下有一個子目錄，名為 `b`，而 `b` 之下又有一個子目錄，名為 `c`。

```
mk_folder a\b\c
```

若文件夾路徑中有空格，則需要把路徑放在兩個雙引號之中，如：

```
mk_folder "a b\c d"
```

6.2 gs_split_pdf.bat 詳解

`gs_split_pdf.bat` 的內容如下：

```

1 @echo off
2 REM use 64-bit ghoscript to split a pdf
3
4 set temp_folder=%1
5 set outfile=%2
6 set infile=%3
7
8 REM the gswinXXc.exe does not prompt the ghostscript window, the ones without the "c"
  ↳do prompt
9
10 gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%temp_folder%/
  ↳%outfile% %infile%

```

此腳本用於把輸入的 PDF 分割為單頁的 PDF。

此腳本接受三個參數。它們如下（按順序）：

1. 輸出的單頁 PDF 的文件夾的路徑
2. 輸出的單頁 PDF 文件的共用文件名加上 “%d”
3. 需要分頁的 PDF 的路徑

此腳本調用的是 64 位的 `ghostscript`，若你安裝的是 32 位的版本則需要把此腳本中的 `gswin64c` 更改為 `gswin32c`。

此處給出一個例子，輸出的文件夾為 `output`（已存在），共用文件名為 `common`，需要分頁的 PDF 為 `pdf_in`。

```
gs_split_pdf output common%d.pdf pdf_in.pdf
```

備註：“%d”

如果不在共用文件名後加上 “%d”，ghostscript 則不會對 PDF 進行分頁。

6.3 pdf_to_svg.bat 詳解

pdf_to_svg.bat 的內容如下：

```

1 @echo off
2 REM use pdf2svg (https://github.com/jalios/pdf2svg-windows) to convert a pdf to
  ↳ individual svgs
3
4 set inputfile=%1
5 set outputfile=%2
6
7 pdf2svg %inputfile% %outputfile% all

```

此腳本用於把 PDF 轉換為 SVG。

此腳本接受兩個參數。它們如下（按順序）：

1. 需要轉換的 PDF 文件路徑
2. 輸出的單頁 PDF 文件的共用文件名加上 “%d”

此腳本調用 pdf2svg 來進行文件的轉換。此處給出一個例子，需要轉換的 PDF 名為 pdf_in，輸出的單頁 PDF 的共用文件名為 pdf_out，輸出的文件夾為 out_svg（已存在）。

```
pdf_to_svg pdf_in.pdf .\out_svg\pdf_out%d.svg
```

備註：“%d”

如果不在共用文件名後加上 “%d”，則只會轉換 PDF 的第一頁。

6.4 pdf_to_png.bat 詳解

pdf_to_png.bat 的內容如下：

```

1 @echo off
2 REM use pdftocairo to convert a pdf into multiple pngs
3
4 set dpi=%1
5 set inputfile=%2
6
7 pdftocairo -r %dpi% -png %inputfile%

```

此腳本用於把 PDF 轉換為 PNG。此腳本會在 CMD 當前所在之目錄創建 PNG 文件。

此腳本接受兩個參數。它們如下（按順序）：

1. 轉換的 PNG 的 DPI，一般為 300 或 600。過高的 DPI 會令轉換過程冗長。一般而言，打印需要至少 300 DPI，一般 600 DPI 足以應付絕大部分情況。
2. 需要轉換的 PDF 的路徑

此腳本調用 pdftocairo 來進行文件的轉換。此處給出一個例子，轉換的 DPI 為 600 像素，需要轉換的 PDF 名為 pdf_in。

```
pdf_to_png 600 pdf_in.pdf
```

備註: `pdftocairo`

`pdftocairo` 會自動把多頁 PDF 轉換為單頁的 PNG, 非常方便。

備註: 為了方便, 用戶可以把 `util` 文件夾的路徑加入到系統環境變量 `Path` 中, 這樣就可以直接調用腳本, 而不需要在命令中指定 `util` 文件夾的路徑。

一步到位

本章在腳本詳解的基礎上，講解怎樣把命令合併到一個腳本中從而簡化 `documentclass` 的配置。

從腳本詳解中可以看出，本教程為了方便排錯，而把命令分到多個腳本之中，逐個擊破，但其實完全是可以把命令全部寫入單個腳本之中，而實現簡化 `documentclass` 的設置的。

本章將會提供這麼做的範例。

7.1 一步轉換成 SVG

本小結將使用 `util` 文件夾中之 `qk_pdf_to_svg.bat` 腳本，以 `mwe` 文件夾中的 `qk_to_svg.tex` 例，展示如何一步轉換到 SVG。

`qk_to_svg.tex` 中的 `documentclass` 的配置如下：

```

1 \documentclass[tikz, convert, convert={outtext=.svg, command=\unexpanded{
2 % 'out_svg' 是用 存放 SVG 的文件
3 % 'out_svg' 是用來存放 SVG 的文件夾
4 % 'out_svg' is the destination folder for SVG files
5 call ../util/qk_pdf_to_svg out_svg \infile\space \outfile\space
6 }]{standalone}

```

其中，`out_svg` 是存放 SVG 的文件夾。

`qk_pdf_svg.bat` 腳本的內容如下：

```

1 @echo off
2
3 set dst_dir=%1
4 set inputfile=%2
5 set outputfile=%3
6
7 if not exist %dst_dir%\NUL mkdir %dst_dir%
8
9 cd /d %dst_dir%
10
11 pdf2svg ../%inputfile% %outputfile% all

```

qk_pdf_to_svg.bat 中的命令，是把 demo_to_svg.tex 中的 documentclass 中的命令整合一，以達到簡化 documentclass 的配置的目的。

7.2 一步轉換成 PNG

本小結將使用 util 文件夾中之 qk_pdf_to_png.bat 腳本，以 mwe 文件夾中的 qk_to_png.tex 例，展示如何一步轉換到 PNG。

qk_to_png.tex 中的 documentclass 的配置如下：

```
1 \documentclass[tikz, convert, convert={outtext=.svg, command=\unexpanded{
2 % 'out_png' 是用 存放 PNG 的文件
3 % 'out_png' 是用來存放 PNG 的文件夾
4 % 'out_png' is the destination folder for PNG files
5 call ../util/qk_pdf_to_png out_png 600 \infile\space
6 }]}{standalone}
```

其中，out_png 是存放 PNG 的文件夾。

qk_pdf_to_png.bat 腳本的內容如下：

```
1 @echo off
2
3 set dst_dir=%1
4 set dpi=%2
5 set intputfile=%3
6
7 if not exist %dst_dir%\NUL mkdir %dst_dir%
8
9 cd /d %dst_dir%
10
11 pdftocairo -r %dpi% -png ../%intputfile%
```

qk_pdf_to_png.bat 中的命令，是把 demo_to_png.tex 中的 documentclass 中的命令整合一，以達到簡化 documentclass 的配置的目的。

7.3 一步轉換成 EMF

本小結將使用 util 文件夾中之 qk_pdf_to_emf.bat 腳本，以 mwe 文件夾中的 qk_to_emf.tex 例，展示如何一步轉換到 EMF。

qk_to_emf.tex 中的 documentclass 的配置如下：

```
1 \documentclass[tikz, convert, convert={outtext=.pdf,
2 command=\unexpanded{
3 % 'out_emf' 是用 存放 EMF 的文件
4 % 'out_emf' 是用來存放 EMF 的文件夾
5 % 'out_emf' is the destination folder for EMF files
6 call ../util/qk_pdf_to_emf out_emf \outfile\space \infile\space
7 }]}{standalone}
```

其中，out_emf 是存放 EMF 的文件夾。

qk_pdf_to_emf.bat 腳本的內容如下：

```

1 @echo off
2
3 set dst_dir=%1
4 set outputfile=%2
5 set intputfile=%3
6
7 if not exist %dst_dir%/NUL mkdir %dst_dir%
8
9 gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%dst_dir%/
  ↳%outputfile% %intputfile%
10
11 cd /d %dst_dir%
12
13 for /r %%i in (*.pdf) do inkscape %%i -M %%~pni.emf
14
15 del /F *.pdf

```

qk_pdf_to_emf.bat 中的命令，是把 demo_to_emf.tex 中的 documentclass 中的命令整合 一，以達到簡化 documentclass 的配置的目的。

7.4 一步轉換成 EPS

本小結將使用 util 文件夾中之 qk_pdf_to_eps.bat 腳本，以 mwe 文件夾中的 qk_to_eps.tex 例，展示如何一步轉換到 EPS。

qk_to_eps.tex 中的 documentclass 的配置如下：

```

1 \documentclass[tikz, convert, convert={outext=.pdf,
2 command=\unexpanded{
3 % 'out_eps' 是用 存放 EPS 的文件
4 % 'out_eps' 是用來存放 EPS 的文件夾
5 % 'out_eps' is the destination folder for EPS files
6 call ../util/qk_pdf_to_eps out_eps \outfile\space \infile\space
7 }]{standalone}

```

其中，out_eps 是存放 EPS 的文件夾。

qk_pdf_to_eps.bat 腳本的內容如下：

```

1 @echo off
2
3 set dst_dir=%1
4 set outputfile=%2
5 set intputfile=%3
6
7 if not exist %dst_dir%/NUL mkdir %dst_dir%
8
9 gswin64c -sDEVICE=pdfwrite -dSAFER -dNOPAUSE -dBATCH -sOutputFile=%dst_dir%/
  ↳%outputfile% %intputfile%
10
11 cd /d %dst_dir%
12
13 for /r %%i in (*.pdf) do inkscape %%i -E %%~pni.eps
14
15 del /F *.pdf

```

`qk_pdf_to_eps.bat` 中的命令, 是把 `demo_to_eps.tex` 中的 `documentclass` 中的命令整合 一, 以達到簡化 `documentclass` 的配置的目的。

總結

本章為此教程之總結。

本教程利用 Latex 中的 `standalone` 包和 Windows 命令和腳本，結合 `pdf2svg`, `pdftocairo`, `inkscape` 和 `ghostscript` 來實現編譯 TEX 文件時，同時把生成的 PDF 轉換成單頁的圖片文件。

本教程講述來如何轉換成 SVG，PNG，EMF 和 EPS 這四種類型的圖片。

如果你需要轉換為 JPG，請參考 EMF 的轉換教程，自行實現。

本項目使用 `git` 進行版本控制。遠端倉庫存放在 GitHub 上。你可以在 [這裡](#) 下載本項目，並提出問題。