

VST (Vim reStructured Text) 簡介

Edward G.J. Lee

2005.12.03.

Author: Edward G.J. Lee
Version: 1.0
License: GNU GPL
Date: 2005.12.03.

使用純文字編輯器也能輸出漂亮、標準的 PDF/HTML/L^AT_EX 文稿格式嗎？這篇文章就簡單的來探討使用率非常高的 Vim 編輯器，在無需深入學習困難的幕後排版指令的前提下，如何做到專業排版輸出的水準，你可以把他當做是一種高水準的簡排系統。

目錄

1 前言	2
2 VST 的安裝及使用	3
2.1 安裝	3
2.2 使用	4
3 基礎語法	5
3.1 章節段落	5
3.1.1 章節標題	5
3.1.2 大綱 (目錄)	6
3.2 註解	6
3.3 字體變化	7
3.4 連結	7
3.4.1 以網址本身來顯示的超連結	7
3.4.2 以其他文字來顯示的超連結	8
3.4.3 引用各章節標題	8
3.5 原文照列 (preformatting)	9
3.6 劃分隔線	9

4 角色扮演 (Roles)	10
4.1 下標字 (:sub:)	10
4.2 上標字 (:sup:)	11
4.3 大字 (:big:)	11
4.4 小字 (:small:)	11
5 腳註 (footnote)	11
5.1 文獻引用	12
6 條列式	12
6.1 英數字條列式	12
6.2 符號條列式	13
6.3 自行定義條列式	14
6.4 特定欄位條列式	15
7 圖形處理	16
8 表格處理	17
9 VST 的進階功能	19
9.1 數學式子	19
9.2 Vim 2html	19
9.3 顏色	20
10 結語	21

1 前言

由於網路的發達，個人網頁，俯拾皆是，部落格的版面也是爭奇鬥豔，各顯所能，大家愈來愈重視版面的排版效果，排版也適時的普及。但是，排版本來就是一門專門的學問，實際的細節非一般人可以直覺的用於平常文章的書寫。

就以較符合大眾口味的幕前排版系統（一般的圖形界面辦公室文書排版處理軟體），要達到細部調整的功能，也並非就是那麼的簡便，而且這類軟體愈做愈大，予人笨重的感覺，有時寫個一、二千字的文章，使用笨重的工具就覺得殺雞用牛刀。

那麼幕後排版呢？他的主流是 [L^AT_EX](#) 及 [DocBook](#)，有接觸過的朋友，沒有被操過、嚇過的大概是少數人。他們的指令非常繁複，記不勝記，學習曲線很陡峭，雖然排版品質很不錯，但不是經常寫文件，或者需要寫論文，而老闆指定要 [L^AT_EX](#) 格式的朋友，大概一般人是太會有學習的欲望。

VST 是 [reStructuredText](#) 的 Vim 版本，是由 [Mikolaj Machowski](#) 移植過來的，並加上了

一些 Vim 本身就有的功能，和 Vim 結合在一起，文件在 Vim 中編輯完成，各種文件格式就可以馬上輸出，例如：HTML、XML、 \LaTeX 及 PDF 等，講求直覺、輕便及短小精悍。reStructuredText 是一種純文字的格式，就像一般編輯器編寫文件一樣，只不過加上了簡單的註記，這些註記除了代表一定意義的排版功能外，它也讓純文字文件更容易閱讀，算是一種不顯得突兀的裝飾，因此他不僅代表結構化，也有美化純文字文件的功能。

這篇文章就是使用 VST 格式所寫的，你可以先參考看看他的表現是否符合你的需求：

- 原純文字格式：<http://edt1023.sayya.org/vim/vst/vstu8.vst>
- HTML 格式：<http://edt1023.sayya.org/vim/vst/vstu8.html>
- \LaTeX 格式：<http://edt1023.sayya.org/vim/vst/vstu8.tex>
- PDF 格式：<http://edt1023.sayya.org/vim/vst/vstu8.pdf>

2 VST 的安裝及使用

2.1 安裝

VST 是一個 Vim script，他的安裝非常簡單，先下載 [vst.vim](#) 把他置放於：

```
$HOME/.vim/plugin
```

目錄下即可。另外，最重要的，他必須和 Vim7（可由 [Vim CVS](#) 中以 vim7 為 module 名來取得）配合才能運作，所以你要安裝一個 Vim7 的版本。我個人是安裝在 /opt/vim7 目錄下，並在編譯的時候，指定 vim 的名稱就是 vim7/gvim7，這樣就不會和系統上的版本搞混了。

如果需要輸出 \LaTeX /PDF 格式，則需要安裝 \TeX 系統，在 UNIX-like 的作業系統通常就是 [te \$\TeX\$](#) 系統。如果你使用 \LaTeX CJK 來處理中文，那麼請下載可以處理中文的版本：

- 可以處理中文 \LaTeX CJK 的版本：<http://edt1023.sayya.org/vim/vst.vim>
- 使用 \LaTeX CJK/dvipdfmx 的版本：<http://edt1023.sayya.org/vim/vst-x.vim>

要注意的是，這個 \LaTeX CJK 的版本是我個人在使用的，所以 \LaTeX CJK 預設的字型是 [cwmu](#)，目前僅支援 UTF-8 編碼文件（使用 \LaTeX CJK 的 UTF8 環境）。而且，HTML 的版本爲了適應筆畫複雜的中文，已將行距加大。除了安裝 CJK 外也要安裝這個字型，否則就要自行去修改所使用的字型。vst-x.vim 則是使用 dvipdfmx 的版本，VST 預設是使用 pdf \LaTeX ，使用 dvipdfmx 的好處是，他製作出來的中文文件有搜尋、拷貝文字內容的功能。

請注意，如果是 [VST Beta 17](#) 以後的版本，他已經分成兩個檔，只要直接按照他的壓縮檔的置放目錄來置放即可，這個部份目前還沒有整理出給 L^AT_EX CJK 使用的版本，但中文 HTML 應該是沒有問題的。

2.2 使用

啓動 vim 時會自動載入 `vst.vim`。如果是使用 `gvim` 的話，會有一個功能選單供使用。命令列使用的話，目前所載入的文件編寫完成（只寫一點點，想輸出來看也是可以），執行：

```
:Vst html ==> 輸出 XHTML 1.0 格式，但檔案未命名
:Vsti html ==> 輸出 XHTML 1.0 格式，檔案是以原主檔名後附上 html
```

以上都會分割出一個視窗並自動載入這個轉換後的檔案。如果 `:Vst` 及 `:Vsti` 後並沒有參數，則預設是輸出 HTML 格式。如果預設就是要輸出同名但附上必要的延伸檔名的話，可以在 `$HOME/.vimrc` 設一行：

```
let g:vst_write_export=1
```

這樣使用 `:Vst` 就可以了，他會自動寫入主檔名和目前編輯中檔名相同，但延伸檔名會自動附上輸出格式，例如 `html/pdf/xml` 等等，但這會將原有的檔案直接覆蓋，不會提出警告。

其他的格式可有：

```
:Vst latex 或 :Vst tex
:Vst pdf
:Vst xml
:Vst rest
```

其中 `rest` 是指原 `reStructuredText` 格式文件，Vim 會移去少部份不相容的註記，並修改成相容的註記。L^AT_EX 及 `pdf` 格式則預設是使用 `pdflatex` 來處理的。要用於中文的話，需要安裝 L^AT_EX CJK 套件，並自行加入必要的 CJK 指令才能正常執行。HTML/XML 格式則由於 Vim 本身能處理中文，所以不會有問題。

如果在輸出 PDF 檔後要立即線上閱覽（預設使用 `xpdf`），那麼可以在 `$HOME/.vimrc` 加一行：

```
let g:vst_pdf_view=1
```

這樣下達 `:Vst pdf` 指令，輸出 PDF 後，就會立刻啓動 `xpdf` 來閱覽。如果要改由 `acroread` 來閱覽，則可以設定：

```
let g:vst_pdf_viewer=acroread
```

3 基礎語法

並不是說 VST 就沒有語法了，他仍然是有一些標記、註記的，這樣才能指示 VST 引擎做適當的轉換處理。而他的後續處理也是要透過其他的排版工具的配合，例如要輸出 PDF/L^AT_EX 格式，系統上要安裝 T_EX 系統，XML 後續的處理，也是需要 DocBook/XML 等相關的工具。這裡我們就先來做簡單的 VST 語法的說明，只介紹我們平常寫文件時最常要用到的，馬上就可以試用看看。較複雜的部份會另外獨立出一個章節來說明。

3.1 章節段落

章節段落除了以章節符號來區分，空一個空白行也是會自動成一個段落。一個空白行和多個空白行的作用是一樣的。斷行也會依文件實際的斷行並配合一行的寬度來處理，縮排亦同。

3.1.1 章節標題

這些標題符號是位於標題名的底下自成一列，而且長度一定要和標題長度一致。

```
===== h1 chapter 章 (大標題)
----- h2 section 節 (中標題)
~~~~~ h3 subsection 小節 (小標題)
+++++ h4 subsection 次小節 (次小標題)
```

例如，章的話，可以寫成：

```
這是第一章
=====
```

這樣 VST 就認得了，我們看純文字文件時也會有顯明、提綱挈領的功用。要非常注意的是空白行，VST 非常依賴空白行來做各種結構的區別，所以，上下一定要空出至少一個空白行出來。

這些章節符號並不一定要固定的，你可以選用以下符號的任何一種，只要整篇文件的章節不同層次使用不同的符號，而且保持一致性就可以了：

! " # \$ % & ' () * + , - . / : ; < = > ? @ [\] ^ _ ` { | } ~

但一般比較建議的是：

= - ' : . ' " ~ ^ _ * + #

目前所編輯的文件是使用了什麼樣的章節符號及他的層次，可在 Vim 中執行以下的指令：

```
:Vst head
```

就可以得知。有時怕忘了各符號所代表的層次，這個指令就很有用了。像這篇文件的章節符號是：

Level	Symbol
h1	*****d
h2	*****
h3	^^^^^^^^
h4	~~~~~

3.1.2 大綱 (目錄)

在編輯 VST 文件時，要查閱目前所編輯的大綱結構（也就是目錄），可使用：

```
:Vst toc
```

來查看，其中後面所顯示的數目字就是所在行，在 Vim 中以 :Number 就可以到達這個標題的所在處。

3.2 註解

兩個英文句點後的文字會變成註解，不會在轉出的 PDF/HTML/XML 格式中出現，但註解裡頭有些內容是 VST 引擎在轉換時的一個指示，所以，在 VST 中，他的註解會有兩種不同的作用，一個是真正的註解，一個是傳達轉換指示給 VST 引擎。

.. 這裡開始是註解。

以下這個雖然和註解一樣不會顯示在 PDF/HTML/XML 格式的檔案，但他和一般的註解的意思不一樣，由 Vim_ 來引用時，會有超連結的功能，可以連結到 <http://www.vim.org>。

.. _Vim: <http://www.vim.org>

3.3 字體變化

由英文星號字元 * 前後所包住的文字，會使用斜體，雙星號 ** 包住的是粗體，雙左單引號 ‘ ‘ 所包住的表示打字機字族。這裡的左單引號是鍵盤左上方波紋號那個鍵。

```
*這會變成斜體 italic*  
**這會變成粗體 bold**  
‘‘這會變成打字機字族 typewriter‘‘
```

這會表現成『這會變成斜體 *italic*』、『這會變成粗體 **bold**』、『這會變成打字機字族 typewriter』。比較不方便及要非常注意的是，這些字體變化符號和其他正常文字間要留個英文空白，否則無法判斷。但這樣一來會多出個空白出來，對中文文件而言不甚美觀，這種情形下可以使用 escape 跳脫的方式，例如：

```
這裡有文字\ *這裡是斜體*\ 這裡又是文字
```

這樣就不會留有空白了，這會表現成『這裡有文字這裡是斜體這裡又是文字』。這種跳脫的方式在 VST 文件中很常會用到，也是使用中文（及其他多字元字節語系）無法避免的。

3.4 連結

連結不僅僅是用於網際網路，本文內部的連結也是可以。連結的表示方式有好幾種，這裡只介紹我個人覺得使用起來較直覺的方式。

3.4.1 以網址本身來顯示的超連結

只要文章中開頭是 http/https/ftp/mailto 的網址在轉換成 HTML/PDF 等格式後就會自動建立超連結。但要注意的是，顯示的字體是使用一般羅馬字族來顯示，但較講究的，這些網址會使用打字機字族來表示，例如以下兩種方式，所使用的字體就不一樣：

- 這是正常羅馬字族：<http://edt1023.sayya.org>
- 這是打字機字族：<http://edt1023.sayya.org>

在較正式的排版場合是使用後者。這也很簡單，只要在網址前後以雙左單引號包住就可以了。這裡要注意的是，這些網址的前後如果還接有其他文字內容，要留個空白，否則判斷上會造成錯誤，尤其是有加字體變化的符號更是如此。

當然，由於網頁瀏覽器的設定問題，上述的字體可能不一定會正確，可參考本文的 PDF 版本就會正確顯示兩種字體上的不同。而且，這只限於網址本身，如果是以文字來代表超連結的話，請使用那個文字原應有的字體、字族。

3.4.2 以其他文字來顯示的超連結

這個就要利用註解的方式來先定義某段文字是網址，然後再來引用這些文字。他的判斷依據及定義原則是前底線定義，後底線引用。這裡舉個例子會比較清楚：

```
.. _Edward: http://edt1023.sayya.org
在文章中引用 Edward_ 就可以了。如果文字中含有空白，引用時要加左單引號：
.. _Edward G.J. Lee: http://edt1023.sayya.org
在文章中要使用 'Edward G.J. Lee'_ 來引用。
```

所以，有必要的話，可以先將常引用到的網址事先定義好，這樣引用時就很方便了，而這些定義在轉換過後的文件內並不會顯示出來，而在純文字的 VST 文件內也有類似註解的功能。那些底線在轉換成其他格式後會消失，就像這樣：[Edward G.J. Lee](http://edt1023.sayya.org)。但引用時前後如果另有接文字的話，要留個英文空白，並請記得如何跳脫空白字元，尤其這些文字是中文的時候。

要知道目前編輯的文件中定義了什麼連結，只要下以下指令就可以知道：

```
:Vst link
```

另外一種以文字來顯示超連結的方式是使用角括號 < > 將網址給括住。例如：

```
'果正札記 <http://edt1023.sayya.org>'_
```

這樣也是可以，要注意的是左單引號及最後的底線，這會表現成『[果正札記](http://edt1023.sayya.org)』。這樣一行就把他給解決掉了，角括號括住的網址不會顯示出來，只有顯示文字的部份，並建立和網址的超連結。

3.4.3 引用各章節標題

我們所定下的章節標題，可以以章節名稱後接一個底線來直接引用連結，例如：

```
章節段落
~~~~~
```

可以由 \ 章節段落_ \ 來引用。

我們可以使用後底線的方式直接引用章節名稱，這在 PDF/HTML 格式會自動建立連結。要注意的是空白的跳脫，因為中文字間空白需要去除，而這個引用的前後又必需空出一個英文空白，所以要把空白像上面一樣的方式給跳脫掉。例如：請參考『[章節段落](#)』這一節的說明，這會直接連結過去。這是一個相當方便直覺的功能。

如果你忘了章節名稱，別急著去找，請記得使用以下這個指令：

```
:Vst toc
```

3.5 原文照列 (preformatting)

原文照列就是寫什麼就顯示什麼，不理會一些排版的標記，這常常會用於程式碼的顯示。這是由 `::` 這個雙冒號所控制，這個符號下要空出一個空白行，而原文照列的內容，通通要縮格，最後一個空白行結束原文照列。例如：

`::`

這裡是原文照列的內容：

使用 `::` 這個符號。而且，內容在書寫時要縮格。
英文的部份會使用打字機字族 `Typewriter`。

原文照列符號 `::` 在轉換成 html/pdf 格式後不會顯示出來，這在實際表現出來會是：

這裡是原文照列的內容：

使用 `::` 這個符號。而且，內容在書寫時要縮格。
英文的部份會使用打字機字族 `Typewriter`。

其實沒有 `::` 這個符號，只要行文時給他縮排，通常 VST 就會把他當做是原文照列，但為了結構上清楚起見，而且避免 VST 誤判，建議加上 `::` 讓文章結構更明確。

原文照列要注意的是，他不會自動折行，這在 HTML 格式，會在原文照列這個方框中形成視窗，有 scroll bar 供滑鼠拖曳，以便閱讀超出行寬的文字，而且這樣也就不會超出瀏覽器的正常閱覽版面的範圍。但這在 PDF 格式則沒有這個功能，會造成過長的文字被截掉了，所以，這個地方在還沒有改善 VST 引擎之前，使用上要非常注意，要視情形手動去斷行。

像以下這個超長的文句，在 HTML 可以處理，在 PDF 則不行，超過版面寬的部份被截掉了：

超長的原文照列文句。一個神聖的目標是很危險的。當一個目標變得神聖時，要達到這個目標的手段往往會變得

這個部份要改善 VST 所引用的 L^AT_EX macro 才能解決，否則 PDF 的部份要手動去修改 tex 文稿。

3.6 劃分隔線

以下的符號都可以用來劃分隔線，也讓純文字內容有分隔出來的效果：

=====

~~~~~

不過，我個人是選個和章節標題不同的符號來使用，以免造成混淆。要注意的是他上下要留一行空白行，他的效果如下：

---

所以，不必想太多，很直覺的就給他畫一整條橫線就對了。所有可用於章節表示的符號都可以拿來畫分隔線，但建議不要和其他地方使用的符號重複。

## 4 角色扮演 (Roles)

由於純文字所能使用的標記、註記符號有限，爲了讓文件內容看起來更結構化，所以會使用特殊的文字來表示標記，它扮演著某個排版功能（角色）。角色扮演的通式是：

:name: 'text'

其中 name 就是所扮演的排版角色的名稱，例如 :sub: 是代表下標字。text 就是你所要書寫的文字，排版功能就會作用在這些 text 上。在 VST 較進階的排版功能裡頭，會不少角色扮演的情况，雖然對純文字文件會有閱讀上的破壞性，但卻是不得已的，只要習慣了就好了，這是純文字格式先天上的限制，幸好，我們寫一般文件的話，這樣子的情况並不是很多，而且有如註解的功能，使文件更具結構化。

### 4.1 下標字 (:sub:)

以硫酸的化學式來說，可以表示成：

H\ :sub: '2' \ SO\ :sub: '4'

表現出來會是『H<sub>2</sub>SO<sub>4</sub>』。當然這和所轉換成的文件格式有關，像 PDF 會有較好的表現，HTML 則因爲先天的限制，表現會比較差一些。

## 4.2 上標字 (:sup:)

例如，2 的  $n$  次方，可以寫成：

```
2\n
```

表現出來會是『 $2^n$ 』。

## 4.3 大字 (:big:)

```
:big:這裡是大字
```

表現出來會是『這裡是大字』。

## 4.4 小字 (:small:)

```
:small:這裡是小字
```

表現出來會是『這裡是小字』。

---

## 5 腳註 (footnote)

前面所談到的註解是不顯示出來的，只是文稿寫作時的說明，是給作者及 VST 引擎看的。這裡的腳註也是註解，但這是要顯示出來給讀者閱讀的。

這裡是一個腳註 [1] (footnote) 的例子。

```
.. [1] 這裡是腳註的真正內容。
```

實際上會表現成『這裡是一個腳註<sup>1</sup> (footnote) 的例子。』，號碼的順序要自行控制好。

也可以使用 # 來代替號碼，這樣 VST 會自動對號入座，例如：

這裡是另一個腳註 [#] (footnote) 的例子。

```
.. [#] 這裡是另一個腳註的真正內容。
```

---

<sup>1</sup>這裡是腳註的真正內容。

實際上會表現成『這裡是另一個腳註<sup>2</sup> (footnote) 的例子。』，號碼的順序 VST 會自行依序控制。

## 5.1 文獻引用

這是腳註的一個特例，表現形式上和腳註差不多，只不過，參考的是文獻資料。

這是一個資料引用 [KDE1989]<sub>1</sub>。

```
.. [KDE1989] Knuth, Donald E., *The TeXbook*, Reading, Massachusetts:  
Addison-Wesley, 1989.
```

這將會表現成：

這是一個資料引用<sup>3</sup>。

當然，這裡會發現，HTML 和 L<sup>A</sup>T<sub>E</sub>X PDF 格式在表現上並不太一樣，而且，嚴格來說，這並不完全符合正式論文的規範，但一般使用應該還是可以的。

---

## 6 條列式

這個使用頻率非常高，也讓文章條列分明，有助閱讀，所以，獨立一個單元來說明。

### 6.1 英數字條列式

最簡單的方式就是直接打上去，整個條列式單元上下要空一行空白行，例如：

1. 第 1 項
  - a. 第 a 小項
  - b. 第 b 小項
  - c. 第 c 小項
2. 第 2 項
3. 第 3 項

---

<sup>2</sup>這裡是另一個腳註的真正內容。

<sup>3</sup>[KDE1989] Knuth, Donald E., *The TeXbook*, Reading, Massachusetts: Addison-Wesley, 1989.

轉換後表現出來將會是：

1. 第 1 項
  - a. 第 a 小項
  - b. 第 b 小項
  - c. 第 c 小項
2. 第 2 項
3. 第 3 項

甚至，更偷懶的方式是以 # 代表英數目字，讓 VST 自行去處理，例如：

- #. 第 1 項
  - a. 第 a 小項
  - #. 第 b 小項
  - #. 第 c 小項
- #. 第 2 項
- #. 第 3 項

其中數目字可以改由英文字母代替，其後的英文句點是不能省略的，這個英文句點也可以由 ) 或 : 來代替。但在使用字母的場合，領頭的字母要標明，這時使用 #，VST 才有辦法自動延續下一個字母，否則預設會把他當成是數字。除了英數字，羅馬記數也是可使用，例如 i, ii, iii, I, II, III 等等也是可以的。

## 6.2 符號條列式

也是和英數字條列式一樣，直接打上去，可以使用符號是 \*、-及+。例如：

- \* 第一項
  - 第一小項
  - 第二小項
- \* 第二項
- \* 第三項

表現出來將會是：

- 第一項

- 第一小項
- 第二小項
- 第二項
- 第三項

英數字及符號也可以混合起來使用。例如：

#. 第 1 項

- 第一點
- 第二點
- 第三點

#. 第 2 項

#. 第 3 項

表現出來將會是：

1. 第 1 項

- 第一點
- 第二點
- 第三點

2. 第 2 項

3. 第 3 項

### 6.3 自行定義條列式

也就是說 item 項是不是單獨有順序的英數字，也不是特殊符號，而是自行定義的文句。他的定義方式也是很直覺，單獨一行做 item，折行後接著是敘述內容段落，而這些段落和 item 間不能有空白行，但各 item 間要有一行空白行。例如：

自行定義 item

這是自行定義的條列式內容。君不見，黃河之水天上來，奔流到海不復回君不見，高堂明鏡悲白髮，朝如青絲暮成雪。

又是一個自行定前 item

人生得意須盡歡，莫使金樽空對月天生我材必有用，千金散盡還復來烹羊宰牛且為樂，會須一飲三百杯岑夫子，丹丘生，將進酒，君莫停與君歌一曲，請君為我側耳聽。

表現出來將會是：

自行定義 `iterm` 這是自行定義的條列式內容。君不見，黃河之水天上來，奔流到海不復回。君不見，高堂明鏡悲白髮，朝如青絲暮成雪。

又是一個自行定義 `iterm` 人生得意須盡歡，莫使金樽空對月天生我材必有用，千金散盡還復來，烹羊宰牛且爲樂，會須一飲三百杯。岑夫子，丹丘生，將進酒，君莫停，與君歌一曲，請君爲我側耳聽。

但由於 HTML 及  $\text{L}^{\text{T}}\text{E}^{\text{X}}$  格式上的差異，自行定義的部份，兩種格式的表現方式可能會有不同，請參考本文的 PDF 及 HTML 兩種格式，互相比較一下。

請注意，這個例子裡頭的中文是一行到底的，這裡要提醒大家，中文不這樣處理的話，將會出現中文字間的空白（斷行的地方會插入一個英文空白），這在英文本來就是字（word）間空白，但在中文就會造成不必要的中文字間空白，這在較具水準的排版上是不及格的。

## 6.4 特定欄位條列式

這經常使用在文章作者、文章名稱及修訂時間等等資訊的列表，目前 VST 認得以下幾種項目：

- Author
- Title
- Date
- Subject
- Version
- License
- Keywords

例如這份文件的開頭就有：

```
:Author: Edward G.J. Lee
:Version: 0.1
:License: GNU GPL
:Date: 2005.12.03.
```

這會以條列式的方式轉換出這些資訊如下：

```
Author:   Edward G.J. Lee
Version: 0.1
License: GNU GPL
Date:    2005.12.03.
```

## 7 圖形處理

圖形引用需要一些條件，例如圖檔名稱，寬高、屬性……等等，因此會需要更多的資訊給 VST 處理。這裡先看實際的一個例子：

```
.. image:: vstmenu.png
   :width: 380
   :height: 253
   :target: self
   :alt: GVim VST 功能選單
   :title: GVim VST 功能選單
```

這表現出來將會是：

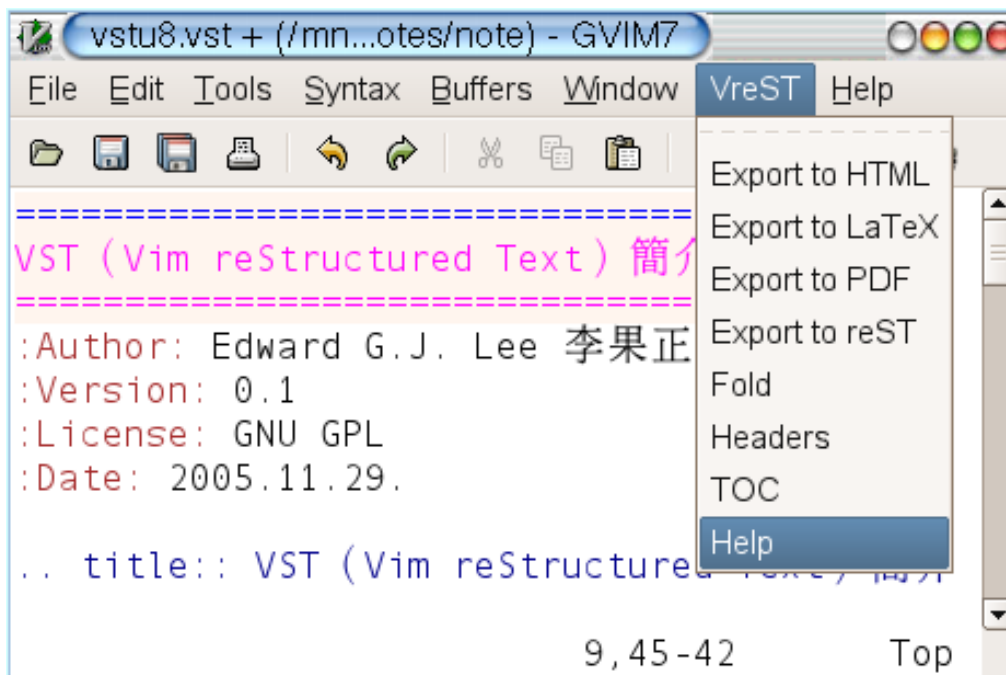


圖 1: GVim VST 功能選單

如果你不想去煩惱圖檔的大小，你可以使用 `:identify:` 來取代指定 `:width:` 及 `:height:`。其後不接參數，就會依原圖大小來顯示，這是呼叫 [ImageMagick](#) 的一個工具程式 `identify` 來得知圖檔大小的資訊，以便處理。如果後面有參數的話，則會以數字的百分比來做縮放後再來顯示。

如果圖檔是存放在網路上，則可以使用 `:target:` 選項來指定要從何處取圖（`:target:` 選項可以有多個同時存在），例如：

```
.. image:: your.png
   :target: http://your.web.site/your.png
   :target: self
```

當然，也可以指定 PATH，這樣會去所指定的目錄取圖。那個 `:target: self` 則會在使用者 click 所顯示的圖後放大原圖大小（僅限於 HTML/XML 格式）。

在 L<sup>A</sup>T<sub>E</sub>X PDF 格式，則會附上標題 (caption)，`:title:` 的部份將會成爲 caption。但這在 HTML 格式則目前尚無法做到。

---

## 8 表格處理

在表格處理方面，VST 的處理方式雖然很直覺，但彈性比較小，而且 L<sup>A</sup>T<sub>E</sub>X 的版本，處理的並不是很完美，有些地方可能需要手動去稍加修改。

基本上，只要直覺的去把他依以下的例子「畫」出來就行了，但要注意前頭不要縮排，否則有可能會被 VST 誤認爲是要原文照列。這裡使用實際例子來說明：

```
=====
=====
日期          行事曆內容大事表
=====
=====
9  十月 2005  - 晉太元中，武陵人，捕魚爲業，緣溪行，忘路之遠近。
                - 忽逢桃花林，夾岸數百步，中無雜樹，芳草鮮美，落英繽紛。
12 十一月 2005 - 復前行，欲窮其林。林盡水源，便得一山。
                - 山有小口，彷彿若有光，便捨船，從口入。
=====
=====
```

表現出來將會是：

| 日期               | 行事曆內容大事表                                                                                                         |
|------------------|------------------------------------------------------------------------------------------------------------------|
| 9<br>十 月<br>2005 | <ul style="list-style-type: none"><li>● 晉太元中，武陵人，捕魚爲業，緣溪行，忘路之遠近。</li><li>● 忽逢桃花林，夾岸數百步，中無雜樹，芳草鮮美，落英繽紛。</li></ul> |

|                   |                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------|
| 日期                | 行事曆內容大事表                                                                                               |
| 12<br>十一月<br>2005 | <ul style="list-style-type: none"> <li>● 復前行，欲窮其林。林盡水源，便得一山。</li> <li>● 山有小口，彷彿若有光，便捨船，從口入。</li> </ul> |

另一個例子：

|      |      |          |      |
|------|------|----------|------|
| 標題一  | 標題二  | 標題三      | 標題四  |
| 欄一列二 | 欄二列二 | 欄三列二     | 欄四列二 |
| 欄一列三 | 欄二列三 | 欄三列三     | 欄四列三 |
| 欄一列四 | 欄二列四 | 這裡可以跨欄設計 |      |

表現出來將會是：

|      |      |          |      |
|------|------|----------|------|
| 標題一  | 標題二  | 標題三      | 標題四  |
| 欄一列二 | 欄二列二 | 欄三列二     | 欄四列二 |
| 欄一列三 | 欄二列三 | 欄三列三     | 欄四列三 |
| 欄一列四 | 欄二列四 | 這裡可以跨欄設計 |      |

由於字型寬度解讀上的問題，原文照列的部份，在你的瀏覽器可能會有線條無法對齊的情形，這些都是要對齊的，你的編輯器必須使用固定寬度（monospace 或 fixed）的字型，否則會無法正確對齊。

這些表格對中文的相容性不是很好，像第二個表格，中文上下行都要留一行空白，否則編譯不會通過，如果你發現 VST 處理時有錯誤，無法產生 HTML 格式檔，請稍加調整中文上下左右的空白。

這在 L<sup>A</sup>T<sub>E</sub>X PDF 的表現也只是差強人意，如果需要精確的講究，可能需自行修改一下所輸出的 L<sup>A</sup>T<sub>E</sub>X 文稿後再手動編譯成 PDF 格式。這是目前 VST 彈性較弱的部份。

---

## 9 VST 的進階功能

如果只是排版一般文件，那麼以上所談到的，大概也就夠用了。但是 VST 也具備引入高階功能的彈性。例如引入真正的 L<sup>A</sup>T<sub>E</sub>X 碼來表現數學式子，但這在 HTML 格式則目前會有困難，只能在 PDF 格式裡頭顯示出來。

### 9.1 數學式子

像以下這樣子的內容則只會出現在 L<sup>A</sup>T<sub>E</sub>X PDF 版本，在 HTML 版本會被忽略：

```
.. raw:: latex
```

以下這個數學式子在 HTML 版本會被忽略：  
$$\int_a^b f(x) dx = \lim_{t \rightarrow b^-} \int_a^t f(x) dx$$

以下這個數學式子在 HTML 版本會被忽略：

$$\int_a^b f(x) dx = \lim_{t \rightarrow b^-} \int_a^t f(x) dx$$

HTML 版本的數學式，目前尚未開始實作。但可插入 raw html 來解決，不過就和 raw latex 一樣，並不是很方便就是了。

### 9.2 Vim 2html

這是 Vim script，原始的 reStructuredText 並不支援。它可以將所指定的檔案類型轉成彩色的 HTML 格式。在 PDF 格式則是一般的原文照列，並不會有彩色。這一般是用於程式碼的輸

出。他的通式是：

```
.. 2html:: 檔案類型 顏色模式
```

檔案類型指的是哪一種程式語言，顏色模式可參考 `$VIMRUNTIME/colors` 目錄下看有哪幾種顏色模式可用，也可依現有的顏色模式，修改成自己需要的顏色模式。例如底下是一個 [Ruby](#) 程式碼的片段，使用 `morning` 的顏色模式：

```
.. 2html:: ruby morning

# check if this file is a TTF or OTF.
def ifTTF(file)
  f = File.new(file)
  tag = f.read(4)
  head = tag[0].to_s + tag[1].to_s + tag[2].to_s + tag[3].to_s
  if (head == '0100') || (tag == 'OTTO')
    return true
  else
    return false
  end
  f.close
end
```

將會表現成：

```
# check if this file is a TTF or OTF.
def ifTTF(file)
  f = File.new(file)
  tag = f.read(4)
  head = tag[0].to_s + tag[1].to_s + tag[2].to_s + tag[3].to_s
  if (head == '0100') || (tag == 'OTTO')
    return true
  else
    return false
  end
  f.close
end
```

### 9.3 顏色

VST 無法直接定義各種顏色，所以可以利用 `raw latex` 及 `raw html` 來自行設定，例如：

```
.. raw:: html
```

```
<font color="#FF0000">這裡是紅色</font>
.. raw:: latex
    \textcolor{red}{這裡是紅色}
```

請注意各行間要有空白行。表現出來將會是：

這裡是紅色

當然，其他更進階的內容也是可以使用這種方式直接導入 HTML 或 L<sup>A</sup>T<sub>E</sub>X 的程式碼。

## 10 結語

VST 並沒有讓 HTML 有分頁的功能，會輸出一個大檔，而且還沒有實作索引的功能。他的定位是在簡排系統，適用於普通文件，將來或許有可能會實作更複雜的功能，也可能不會。

MathML 的支援也完全沒有實作，必須自行插入 raw html/xml code，到時會不會實作及使用哪一種方式實作，還在未定之天。但很明顯的，只要很直覺的打字就可以輸出品質不錯的 HTML/PDF，這在 Vim 而言，大大的補足了這個純文字編輯器的功能。

事實上 VST 雖然源自於 reStructuredText，但他並沒有完全實作 reStructuredText 的全部，只是實作了基本的功能而已，VST 的作者也坦承，reStructuredText 的發展腳步太快，他無法完全跟得上，而且，他對一些特殊的 Python 程式碼的部份也沒有興趣。

使用 VST 的好處是，只要你以 Vim 做為你的主力編輯器，那麼就可以很輕鬆的有 VST 的功能。缺點，當然除了以上所談到沒有實作的部份外，那就是你完全不使用 Vim 的話，那當然就無法使用 VST 了，而且 Vim 7 要成為穩定的版本，可能還需要等待一段時間。這時你可以選擇其他更專業的排版系統來配合，或使用 Python 版的原始 reStructuredText。