程式人 月刊 雜誌

Programmer

讀書做善事、寫書做公益 – 歡迎程式人認養專欄或捐出您的網誌

參考價： NT 50 元，如果您喜歡本雜誌，請將書款捐贈公益團體

羅慧夫顱顏基金會 彰化銀行 (009) 帳號： 5234-01-41778-800

012378

愛心條碼

程式人雜誌

開放公益出版品

2013 年 **4** 月號

本期內容

授權聲明

程式人短訊

出版短訊-**Markdown** 寫作格式

Markdown 是一種由 John Gruber 和 Aaron Swartz 所創造的超簡單型標記語言，目前在網路越來越多的人使用這種格式進行寫作，其用途有點像維基百科 所使用的 mediawiki 格式，但是設計理念卻有很大的不同。

John Gruber	Aaron Swartz
 Gruber at South by Southwest 2009 **Occupation** Columnist **Citizenship** United States **Subjects** Design, Technology, Apple Inc.	 Swartz at 2009 Boston Wikipedia Meetup

Markdown 的設計理念有點像文書排版軟體 (例如 MS. Word, LibreOffice Writer) 這種「所視即所得」的想法，但卻是純文字形式的寫法，也就是讓你寫出來的文字檔就與呈現時的畫面 差不多，這樣的設計讓您在寫作時就很容易感受排版出來的結果，這種作法正是 Markdown 令人愛不釋手 的原因。

為了讓讀者 實際體會 Markdown 的所視即所得效果，我們將本文的部分原始碼列於此處，這樣讀者應該就可以很容易的透過對照看出這些標記的效果了

```
## 出版短訊：markdown 寫作格式

[Markdown] 是一種由 [John Gruber] 和
[Aaron Swartz](http://zh.wikipedia.org/wiki/Aaron_Swartz)
所創造的超簡單型標記語言，目前在網路越來越多的人
使用這種格式進行寫作，其用途有點像維基百科
所使用的 mediawiki 格式，但是設計理念卻有很大的不同。

John Gruber                         Aaron Swartz
----------------------------     --------------------------------
![](../img/JohnGruber.jpg)         ![](../img/AaronSwartz.jpg)

...

為了讓讀者 *實際體會* Markdown 的所視即所得效果，
```

```
我們將本文的原始碼列於此處，這樣讀者應該就可
以很容易的透過對照看出這些標記的效果了。

...

如果您想要進一步瞭解 markdown 的語法，
請參考 <http://markdown.tw/> 這個寫得很好的 markdown 語法介紹。

[John Gruber]:http://en.wikipedia.org/wiki/John_Gruber
[Markdown]:http://zh.wikipedia.org/wiki/Markdown
```

如果您想要進一步瞭解 markdown 的語法，請參考 http://markdown.tw/ 這個寫得很好的 markdown 語法介紹。

說明：必須注意的是，Markdown 當中並沒有設計表格的語法，所以本文中的表格語法是 pandoc 軟體採用「所視即所得」精神所自行延伸的語法。

(本文由陳鍾誠修改自維基百科，來源為 http://zh.wikipedia.org/wiki/Markdown)

軟體短訊-**Pandoc** 格式轉換系統

在 Web 盛行之後，使用簡易文字的方式撰寫網頁的需求就一直存在，於是產生了像 wiki 這樣的格式， 相對的也就需要一些工具將這些格式轉換成網頁。

維基百科就是這樣一個例子，他們透過 Mediawiki 這樣的格式撰寫文章，然後維基百科的網站就會用 程式將這種簡易的格式轉換成網頁，呈現給使用者看。

Wiki 類型的簡易書寫格式，除了 Mediawiki 之外，還有越來越多從 markdown 衍生出來的語法，像是 ReStructureText (RST), Textile 等等。

在上一篇短訊中，我們介紹了 Markdown 這種很容易撰寫與閱讀的格式，當然我們也就需要一種程式， 可以將 markdown 文件轉換成網頁，pandoc 正是這樣的一個工具程式。

Pandoc 不只可以將 Markdown 轉換成網頁，還可以轉換成 epub, latex, doc, odt 等格式的電子書， 也可以在各種簡易書寫格式之間轉換 (例如 markdown 轉 mediawiki)，因此 pandoc 也被稱為文件格式 轉換的瑞士萬用刀。

Pandoc 的使用很容易，以下是一些 pandoc 的使用範例，其參數的用法與 c 語言編譯器 gcc 很像。

```
pandoc input.md -o output.html              # markdown 轉 html

pandoc -s input.md -o output.tex            # markdown 轉 latex

pandoc -s --webtex input.md -o output.html  # 有數學式可加上 --webtex, --mathml 或 -
```

```
pandoc -s --toc -c ../css/pmag.css -B header.htm -A footer.htm input.md -o output.htm
```

Pandoc 會自動根據輸入輸出檔的附檔名進行辨認，因此不需要特別指定輸入輸出檔的格式 (當然也可以強制指定格式)。

參數 -s 代表 --standalone，也就是要產生獨立可顯示的輸出檔時使用的，像是單獨的網頁 html，或者 epub 等都會 加上這個選項。

Pandoc 也支援一些 markdown 的延伸語法，像是 latex 數學式可用 $ ... $ 這樣的方式包裹住， 例如 $\int_0^{\infty} f(x) dx$ 的呈現結果會是 $\int_0^{\infty} f(x)dx$，但是要用數學式必須加上 --webtex, --mathml 或 --mathjax 等參數，以便選擇數學式要用哪種顯示方式呈現。

在上述範例中的最後一句，--toc 代表要產生目錄 (Table Of Content, TOC)，而 -c ../css/pmag.css 則是要套用該 CSS 樣式，-B header.htm 代表要在輸出檔案前補上 header.htm 作為檔頭，而 -A footer.htm 則代表 要在輸出檔案尾端補上檔尾 footer.htm。

Pandoc 的官方網站的網址如下：

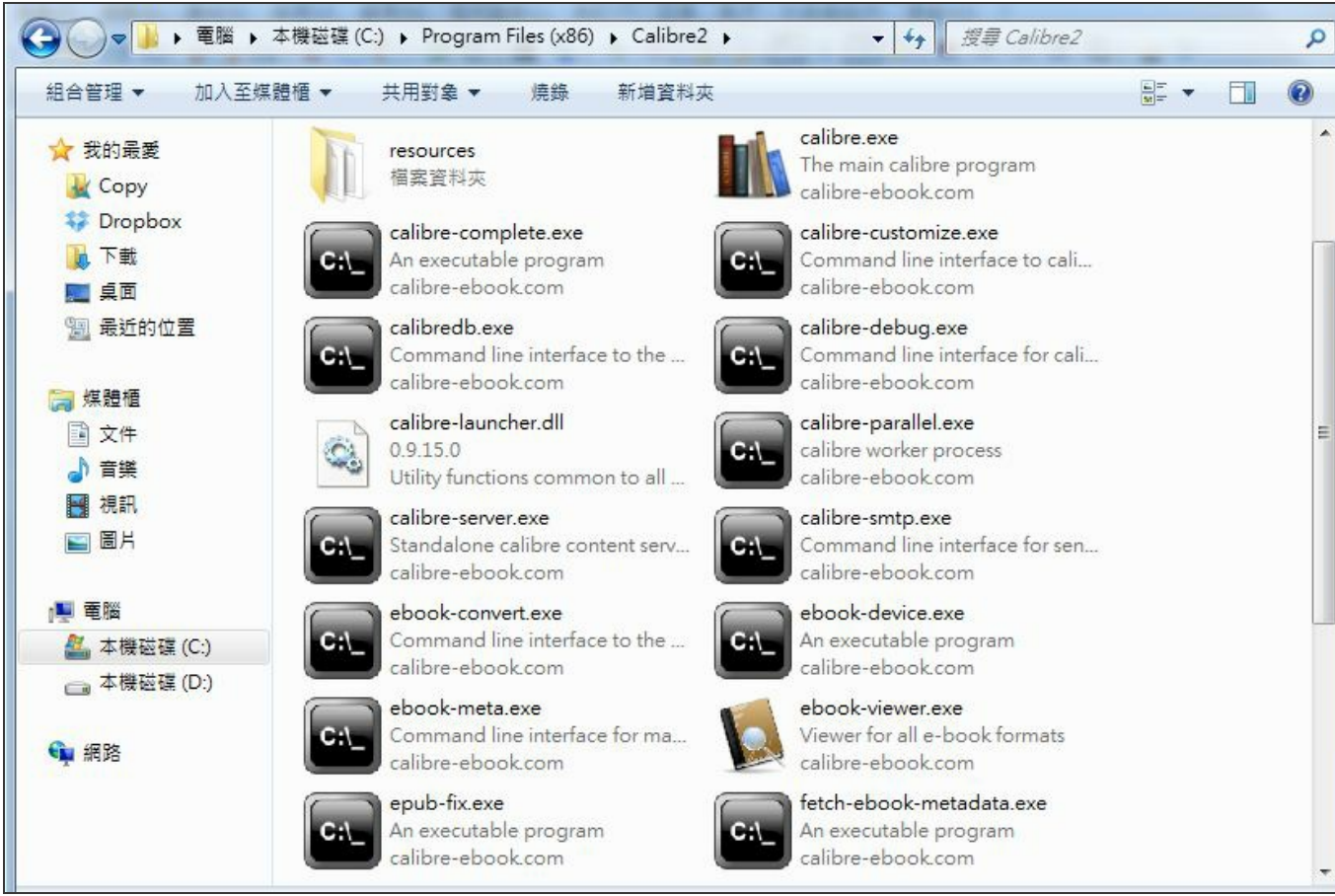
- http://johnmacfarlane.net/pandoc/

您可以從其中的 README 取得詳細的 使用說明，或者直接參考 demos， 以範例的方式學習 pandoc 的用法。【本文由陳鍾誠撰寫】

軟體短訊-**Calibre** 電子書管理轉換系統

Calibre 是一個用來管理並且轉換電子書格式的系統，您可以匯入電子書到 Calibre 當中進行管理與閱讀，也可以將電子書轉換成其他格式，以便放到某些平台上去閱讀。

Calibre 具有視窗型的介面，但是筆者並不常使用，筆者較常用的是 Calibre 當中所附的一個命令列程式，稱為 ebook-convert，下圖顯示了筆者電腦中這個程式所在的位置。



Ebook-convert 可以將電子書的格式轉來轉去，例如將 epub 轉換為 pdf 檔案輸出，以下是該程式支援的 輸出入檔案格式：

> Input Formats: CBZ, CBR, CBC, CHM, DJVU, EPUB, FB2, HTML, HTMLZ, LIT, LRF, MOBI, ODT, PDF, PRC, PDB, PML, RB, RTF, SNB, TCR, TXT, TXTZ
>
> Output Formats: AZW3, EPUB, FB2, OEB, LIT, LRF, MOBI, HTMLZ, PDB, PML, RB, PDF, RTF, SNB, TCR, TXT, TXTZ

您可以看到 ebook-convert 所支援的電子書格式真的非常完整，請進一步閱讀以下網頁以瞭解其用法。

- http://manual.calibre-ebook.com/cli/ebook-convert.html

事實上、程式人雜誌自從第二期之後，就是使用 pandoc 將 markdown 轉換為 html 與 epub 格式，然後 再用 Calibre 的 ebook-convert 程式，將 epub 電子書轉換為 PDF 格式，以便讓讀者能方便的閱讀。

或許有些曾經用過 pandoc 的讀者會有疑問，為何不用 pandoc 直接將 markdown 轉換為 pdf 呢？

原本、筆者曾經這樣做過，但是這樣作在英文上沒有問題，但在中文上就無法成功轉換了。為了解決 pandoc 中文轉換的問題，筆者也曾經撰寫過 latex 樣版，讓 pandoc 順利的將 markdown 轉換為 latex 之後再用 MiKTeX 將 latex 轉換為 pdf 文件。雖然最後可以運作，但是仍然有些無法令人滿意的問題，所以後來筆者才找到了 Calibre 中的 ebook-convert，來完成 epub 轉為 pdf 的動作。

Calibre 官方網站的網址為 http://calibre-ebook.com/ ，您可以從網站中取得進一步的資訊。【本文由陳鍾誠撰寫】

軟體短訊-**Make** 專案建置工具

Make 是 GNU 組織所釋出的老牌專案建置工具，通常會與 gcc 一同安裝，用來建置 c 語言的專案， 但是後來很多人也用 Make 來管理其他語言的專案，甚至有很多語言也發展出類似 make 的專案建置工具， 像是 Java 的 Maven 與 Ruby 的 Rake 等等。

Make 的預設建置對像為該資料夾下名稱為 Makefile 的檔案，因此您只要撰寫好 Makefile，然後執行指令 make 就可以開始建置整個專案了。

由於 Makefile 當中可以呼叫任何的命令列程式，因此並不受限於 C 語言編譯建置時使用，您可以把 Make 當成是更好用的批次檔來用。

事實上、程式人雜誌從第二期以後就是用 make 建置整本電子書的，以下是程式人雜誌的專案建置檔。

```
PANDOC = pandoc -s --webtex
HTML  = home.html license.html message1.html message2.html message3.html \
message4.html people1.html people2.html video1.html article1.html \
article2.html article3.html article4.html info.html
MD   = license.md message.md message1.md  message2.md  message3.md message4.md \
people.md people1.md  people2.md video.md video1.md article.md article1.md \
article2.md article3.md article4.md info.md reflink.md
PARG = -c ../css/pmag.css -B header.htm -A footer.htm
```

```
PEPUB = --toc --epub-metadata=metadata.xml --epub-stylesheet=../css/pmag.css
EARG = --margin-top=16 --margin-bottom=16 --margin-left=20 --margin-right=20 \
--pretty-print --base-font-size=9 --font-size-mapping="7, 8, 9, 12, 14, 16, 20, 24"
RM = rm -f

.PHONY: all epubipad pdfipad pdfA4

all: $(HTML)

epubA4:
    $(PANDOC) $(PEPUB) --epub-cover-image=../img/coverA4.png toc.md $(MD) -o ../book/

pdfA4: epubA4
    ebook-convert ../book/A4.epub ../book/A4.pdf $(EARG) --paper-size=a4

epubipad:
    $(PANDOC) $(PEPUB) --epub-cover-image=../img/cover.jpg toc.md $(MD) -o ../book/ip

pdfipad: epubipad
    ebook-convert ../book/ipad.epub ../book/ipad.pdf $(EARG) --output-profile=ipad3
```

```
html: $(HTML)

shtm:
    $(PANDOC) --toc $(MD) -o ../book/pmag.html $(PARG)

%.html: %.md
    $(PANDOC) $< reflink.md -o ../htm/$@ $(PARG)

clean:
    ${RM} ../htm/*.*
```

在以上的專案建置檔中，我們使用了以下 pandoc 指令將 markdown 轉換為一頁一頁的 HTML，以建立網頁版的程式人雜誌。

```
%.html: %.md
    $(PANDOC) $< reflink.md -o ../htm/$@ $(FLAGS)
```

然後、我們也使用了以下指令呼叫 pandoc 將所有檔案合併轉換為 epub 電子書。

```
epubipad:
    $(PANDOC) $(PEPUB) --epub-cover-image=../img/cover.jpg toc.md $(MD) -o ../book/ip
```

最後、我們使用了 calibre 的 ebook-convert 將 epub 電子出轉換為 pdf 檔案，

```
pdfipad: epubipad
    ebook-convert ../book/ipad.epub ../book/ipad.pdf $(EARG) --output-profile=ipad3
```

以下是我們程式人雜誌 2013 年 4 月號的建置過程。

```
D:\Dropbox\Public\pmag\201304\source>make html
pandoc -s --webtex  home.md reflink.md -o ../htm/home.html -c ../css/pmag.css -B
 header.htm -A footer.htm
pandoc -s --webtex  license.md reflink.md -o ../htm/license.html -c ../css/pmag.
css -B header.htm -A footer.htm
pandoc -s --webtex  message1.md reflink.md -o ../htm/message1.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  message2.md reflink.md -o ../htm/message2.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  message3.md reflink.md -o ../htm/message3.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  message4.md reflink.md -o ../htm/message4.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  people1.md reflink.md -o ../htm/people1.html -c ../css/pmag.
```

```
css -B header.htm -A footer.htm
pandoc -s --webtex  people2.md reflink.md -o ../htm/people2.html -c ../css/pmag.
css -B header.htm -A footer.htm
pandoc -s --webtex  video1.md reflink.md -o ../htm/video1.html -c ../css/pmag.cs
s -B header.htm -A footer.htm
pandoc -s --webtex  video2.md reflink.md -o ../htm/video2.html -c ../css/pmag.cs
s -B header.htm -A footer.htm
pandoc -s --webtex  article1.md reflink.md -o ../htm/article1.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  article2.md reflink.md -o ../htm/article2.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  article3.md reflink.md -o ../htm/article3.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  article4.md reflink.md -o ../htm/article4.html -c ../css/pma
g.css -B header.htm -A footer.htm
pandoc -s --webtex  info.md reflink.md -o ../htm/info.html -c ../css/pmag.css -B
 header.htm -A footer.htm

pandoc -s --webtex  --toc --epub-metadata=metadata.xml --epub-stylesheet=../css/pmag.
  --epub-cover-image=../img/cover.jpg toc.md license.md message.md message1.md  messa
  message3.md message4.md people.md people1.md  people2.md video.md video1.md video2.
```

```
  article.md article1.md article2.md article3.md article4.md info.md reflink.md -o ..
ebook-convert ../book/ipad.epub ../book/ipad.pdf --margin-top=16 --margin-bottom=16
--margin-left=20 --margin-right=20 --pretty-print --base-font-size=9
--font-size-mapping="7, 8, 9, 12, 14, 16, 20, 24"  --output-profile=ipad3
1% 將輸入轉換為HTML格式...
InputFormatPlugin: EPUB Input running
on D:\Dropbox\Public\pmag\201304\book\ipad.epub
Parsing all content...
COLOR_VALUE: No match in Choice(HEX color, Named Color, Sequence(FUNCTION, Choice(una
number, percentage), Sequence(comma, Choice(unary +-, number, percentage)), end FUNC
Sequence(FUNCTION, Choice(unary +-, number, percentage), Sequence(comma, Choice(unary
number, percentage)), end FUNC ")")): ('HASH', u'#\ufeff\ufeff000080', 66, 18)
COLOR_VALUE: No match in Choice(HEX color, Named Color, Sequence(FUNCTION, Choice(una
number, percentage), Sequence(comma, Choice(unary +-, number, percentage)), end FUNC
Sequence(FUNCTION, Choice(unary +-, number, percentage), Sequence(comma, Choice(unary
number, percentage)), end FUNC ")"))
34% 正在對電子書籍進行轉換...
Merging user specified metadata...
Detecting structure...
Flattening CSS and remapping font sizes...
Source base font size is 12.00000pt
```

```
Removing fake margins...
Cleaning up manifest...
Trimming unused files from manifest...
Creating PDF Output...
67% 執行 PDF Output 外掛程式
71% Rendered cover.xhtml
75% Rendered title_page.xhtml
79% Rendered ch1.xhtml
83% Rendered ch2.xhtml
87% Rendered ch3.xhtml
91% Rendered ch4.xhtml
95% Rendered ch5.xhtml
100% Rendered ch6.xhtml
Rendered PDF in 4.316 seconds:
PDF output written to D:\Dropbox\Public\pmag\201304\book\ipad.pdf
將輸出儲存到    D:\Dropbox\Public\pmag\201304\book\ipad.pdf
```

透過這樣的方式，我們用很「程式人」的方法，編輯了一份雜誌。【本文由陳鍾誠撰寫】

程式人介紹

Linux 創建者-Linus Torvalds

Linus Torvalds 這位仁兄我想不用介紹，大家都知道他做了甚麼事，因為有一個作業系統幾乎就是用他名字命名的，Linu-X，簡稱 Linux。

Torvalds in 2002[1]

Linus Benedict Torvalds 於 1969年12月28日出生於芬蘭赫爾辛基市，擁有美國國籍。他除了創造並主導 Linux 核心 的開發之外，還發展出 Git 這個廣為世人使用的版本管理系統。如果諾貝爾獎設立程式領域的項目，那他肯定是得獎者之一。

Torvalds 畢業於赫爾辛基大學計算機科學系，1997年至2003年在美國加州矽谷任職於全美達公司（Transmeta Corporation）參與該公司晶片的code morph技術研發。後受聘於開源碼發展實驗室（OSDL : Open Source Development Labs, Inc），全力開發 Linux 核心，並任職於 Linux 基金會。

托瓦茲與妻子托芙（Tove，一位六次芬蘭前女子空手道冠軍）育有三名孩子。

Torvalds 在大學時代，由於想在 x86 個人電腦上使用類似 UNIX 的系統，但是卻找不到這樣的作業系統。只看到一個由 Andrew Stuart Tanenbaum 這位「荷蘭阿姆斯特丹自由大學的計算機科學教授」所寫的小型 MINIX 作業系統可用，於是他在 研究了 MINIX 之後，於 1991 年開始就逐步改造了 MINIX，最後全部換成了自己的程式碼，並將相關訊息放到 Usenet 新聞群組comp.os.minix 上，標題如下所示：

> Linus Benedict Torvalds (5 October 1991). "Free minix-like kernel sources for 386-AT".

從那時候開始，Linux 就迅速發展，Linux 站在 GNU gcc 這樣的開發工具上，逐步壯大起來，並且吸引了越來越多的程式人員 為此平台開發程式。

Torvalds 開發的是 Linux Kernel，也就式作業系統的核心部分，然後許多人將此核心連同一大群軟體，包裝成許多的分支，像是 RedHat, Ubuntu, Fedora, CentOS, Debian 等等。

後來、連商業公司也開始進來了，基於 Linux 的 Android 作業系統在手機上取得了重大的成功，這讓 Google 這樣的公司 進入了作業系統的領域，並結合整個電子產業的硬體公司，像是 HTC、三星、Motorola、ASUS 等，占領了智慧型手機 的大部分市場。

Linux 改造了整個電腦工業，其創造者 Torvalds 扮演著重要且關鍵的角色。

Torvalds 的另一個重要貢獻是創造並主導了 Git 這個版本管理系統的開發，現在 Git 已經凌駕 CVS , SVN 與 Mercurial，成為 程式人員最常使用的版本管理系統，著名的 github 網站就提供了這樣一個平台，讓全世界的人可以將原始碼上傳並且透過 git 進行開發與合作，連這本「程式人雜誌」也是透過 github 發行並提供給讀者下載的。

Torvalds 的驚人貢獻，來自學生時代的一個想法，並將這個想法付諸行動，成就了不凡的一生。

參考文獻

- Wikipedia:Linus_Torvalds
- 維基百科:林納斯·托瓦茲
- Mac OS X 背后的故事（二）——Linus Torvalds的短视

【本文由陳鍾誠取材並修改自維基百科】

Markdown 與 RSS 的創造者-Aaron Swartz

圖、Aaron Swartz 2008 年 12 月 13 日在 Creative Commons event 的照片

Aaron Swartz 是個程式領域的天才，但是他的人生，卻以悲劇收場。

Aaron Swartz 於 1986 年 11 月 8 日出生美國芝加哥的伊利諾伊州一個猶太家庭。他的父親本身就是程式領域的創業者， 這讓他從很小就開始接觸程式與網路，並於13歲時就贏的了 ArsDigita Prize 比賽的冠軍。

14 歲時，斯沃茨加入了 RSS-DEV 組織，與夥伴們一同創造出了 RSS 規格，後來他積極參與各種網路規

格制定組織，像是為 W3C 撰寫 RFC 以定義 RDF 及 XML 等規格，以及參與創作共用 Creative Commons (CC) 組織，並成為了 CC 創辦人 Lawrence Lessig 指導的研究員。

如果您仔細觀察 Aaron Swartz 的程式、文章與經歷，您會發現他的生命幾乎完全投入在數位出版領域，而且有非常大個貢獻。

舉例而言、除了參與 RSS, RDF, XML 規格的制定，並參與 CC 組織之外，Aaron Swartz 還與 John Gruber 共同創造了 Markdown 這個影響深遠的文件格式，這個格式已成為 github, stackoverflow 等網站的主要書寫格式，並且有眾多工具 pandoc, kitabu 被創造出來將 markdown 用在出版領域，而且也有像 leanpub 這樣的出版網站使用此種格式進行出版。

另外、他還創造出 web.py 這個以 Python 語言寫成的網站框架，用來作為出版平台。並且改寫了 Reddit.com 這個訊息分享網站的程式，用程式讓訊息的分享更為容易。

正是因為對數位出版的熱愛，讓 Aaron Swartz 更感覺到當今商業公司把持了大量的出版利益，並且透過法律體系抑制版權開放的運動，因此他寫下了「游擊隊開放存取宣言」(Guerrilla Open Access Manifesto)，並於 2010 年成立了 DemandProgress.org 網站，反對 SOPA/PIPA 這些網際網路審查法案。

接著、他更在 2010 年 9 月，在 MIT 校園內透過網路自動大量下載了 480 萬份 JSTOR 的期刊論文與相關文件，期望讓這些學術論文公開自由的被存取。接著在 2011 年 1 月 6 日時，他被聯邦調查局逮捕了，並控以「電信詐欺 (wire fraud)、電腦詐欺 (computer fraud)、從受保護電腦非法取得資訊、故意毀壞受保護電腦」等四項罪名。

後來這些罪名陸續增加，從 4 項變為 13 項，並且檢察官威脅他要以刑事罪起訴，這讓他非常沮喪。

2013 年 1 月 11 日時，他在住處自殺身亡。

Aaron 死了，年僅 26 歲！

參考文獻

- Aaron Swartz 的個人網站 -- http://www.aaronsw.com/
- 紀念 Aaron Swartz 的網站 -- http://www.rememberaaronsw.com/
- Inside 硬塞的網路趨勢觀察:早逝的天才，網路神童與資訊自由鬥士:Aaron Swartz
- PanSci 泛科學:美國司法部如何追殺資訊自由化推手 Aaron Swartz
- OpenFoundary:網路運動者 Aaron Swartz 自殺身亡，各方紛紛表達哀悼紀念

後記

按理說，那些學者寫的論文，應該是全人類共有的文明資產才對，但是為何會變成幾家出版商所控制的私有資產呢？ 非學術界的人通常無法理解此種情況，而學術界的人，則是理解之後仍然將論文交給出版商，繼續為他們創造私有資產。

目前的情況是、學者們所寫的論文投稿到期刊之後，商業權將移轉到期刊公司，於是那些未付費就下載這些論文的行為 就都成了侵害私有財產權的行為。

更糟糕的是，連那篇論文的作者，也不能再出版自己的論文，有時連將自己的論文放上網路都成了非法行為。

對這個問題有興趣的朋友可以參看以下的幾篇文章。

- 科技橘報：「封鎖知識」VS.「釋放知識」－－致 Aaron Swartz
- 洪朝貴, 學者的研究成果淪為出版商的生財工具－談學術作品的開放近用
- 洪朝貴, 學者的智慧, 期刊的財產, 圖書館的業績...或是負擔?
- 陳鍾誠, 為台灣教育界投下一顆震撼彈！(續篇)

【本文由陳鍾誠取材並修改自維基百科】

程式人頻道

看影片學 **markdown** 編輯出版流程

在本期的短訊中，我們介紹了 markdown, pandoc, calibre ebook-convert, make 等工具，這些文章列表如下：

- 出版短訊-Markdown 寫作格式
- 軟體短訊-Pandoc 格式轉換系統
- 軟體短訊-Calibre 電子書管理轉換系統
- 軟體短訊-Make 專案建置工具

當您閱讀了以上文章之後，就可以瞭解「程式人雜誌」的編輯方法，並且知道如何用 markdown 格式出版電子書了。

為了讓讀者能有更「真實」的體認，筆者將這些編輯流程錄影下來，放到了 YouTube 上，請點選下列連結以觀賞該影片。

- 看影片學 markdown 編輯出版流程-以程式人雜誌編輯為例 -- http://youtu.be/C7cKrXvTcac

透過這種方式，我們可以採用非常「程式人的方法」，同時出版各種格式的電子書，讓您的身份從程式人轉變為作家。【本文由陳鍾誠撰寫】

看影片學 C# 遊戲程式-使用 **Window Forms**

從本月開始，Gary Lin 將幫我們錄製一系列的影片，從簡單的 Window Forms 程式開始，逐步帶我們進入遊戲設計的領域，目前這系列的影片已經有四集了，編輯將其列表如下：

影片名稱	網址	說明
C# 遊戲程式設計 (00)	http://youtu.be/U8muJ8k1S-U	兩球互相追逐
C# 遊戲程式設計 (01)	http://youtu.be/0zkJLEp0sYY	很多紅球追一個籃球
C# 遊戲程式設計 (02)	http://youtu.be/0HzjzIYYiR4	碰撞處理
C# 遊戲程式設計 (03)	http://youtu.be/U8muJ8k1S-U	多個籃球與紅球

(本文中的影片與教學由 Gary Lin 主講與提供，編輯者為陳鍾誠)

程式人文集

Arduino入門教學(4) – 控制 LED 燈光亮度 (作者：Copper Maa)

Arduino 筆記 - Lab3 控制 LED 燈光亮度

實驗目的

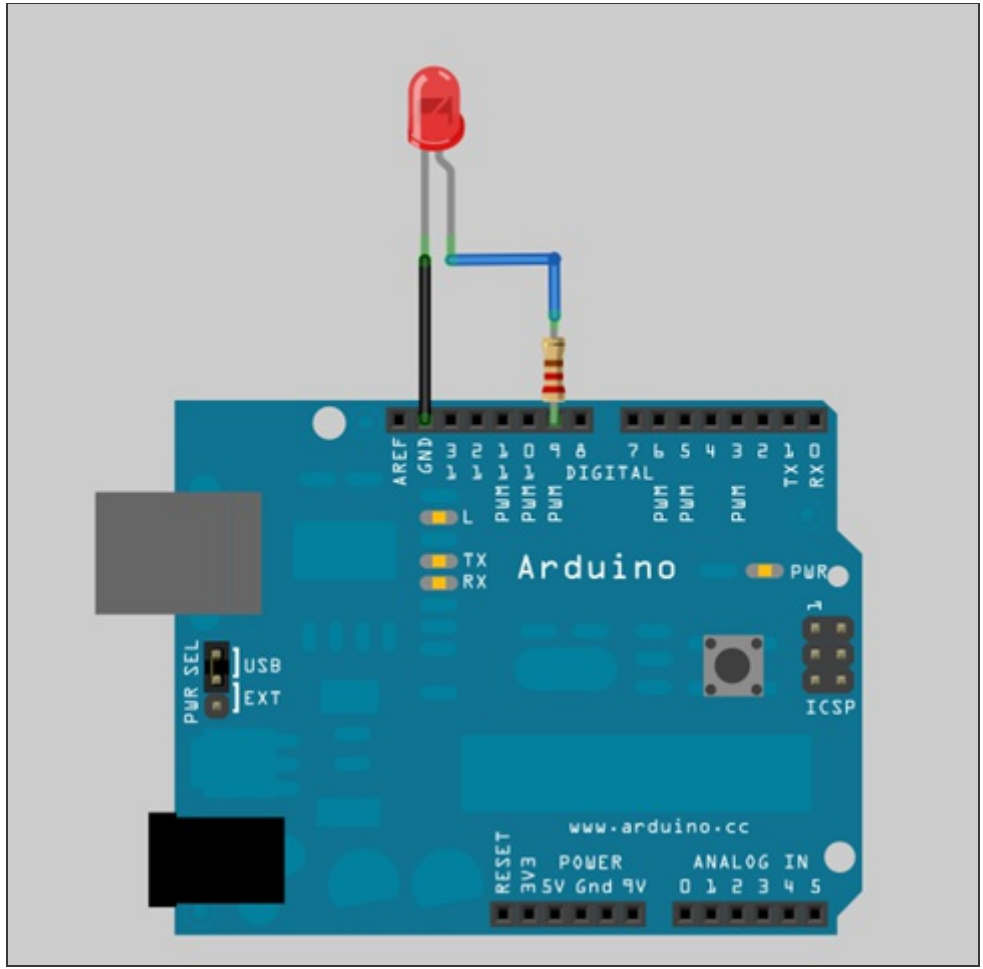
利用 PWM (Pulse Width Modulation, 脈衝寬度調變) 控制 LED 燈光亮度。

材料

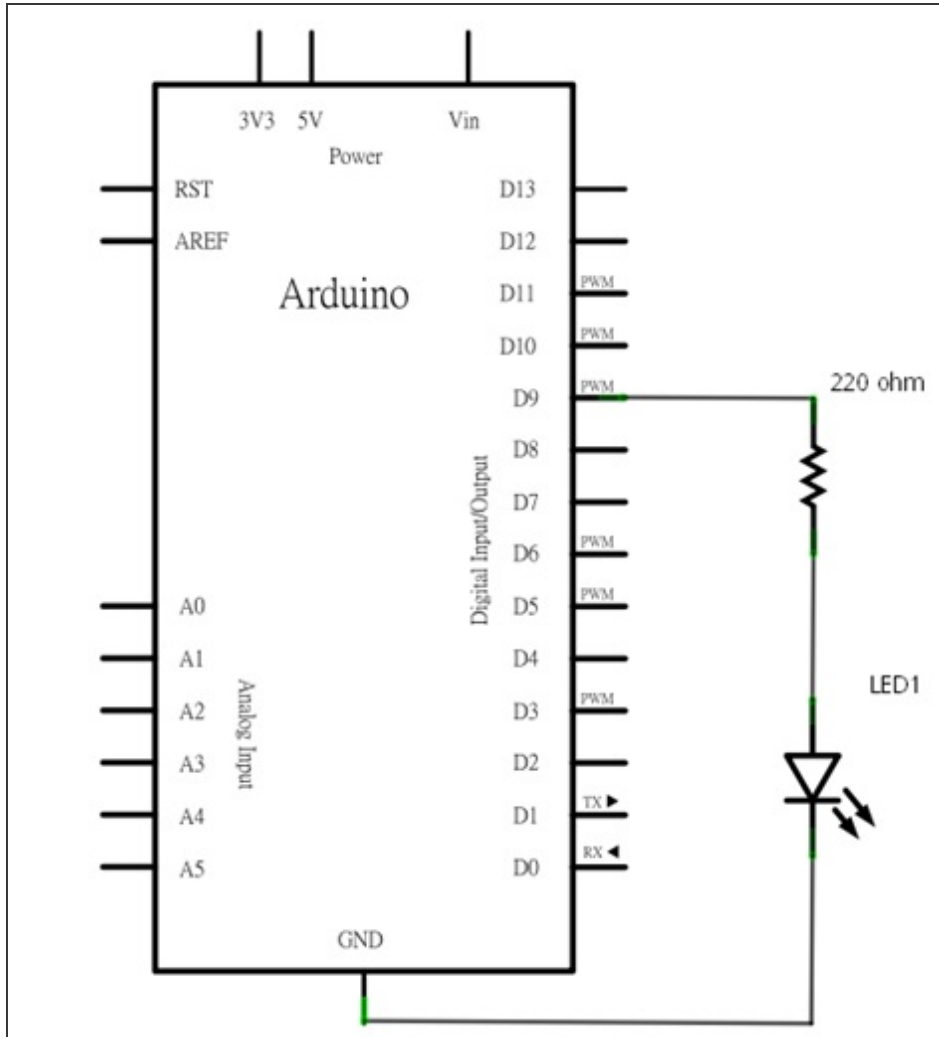
- Arduino 主板 x 1
- LED x 1
- 220 ohm 電阻 x 1
- 單心線 x N

接線

- LED 接到 pin9 和 GND，長腳(陽極)串接一顆 220 ohm 電阻到 pin9，短腳(陰極)直接接到 GND


AREF
GND
13 12 11 10 9 8
7 6 5 4 3 2 1 0
DIGITAL
PWM PWM PWM PWM PWM
TX
RX
L
TX
RX
Arduino
PWR
PWR SEL
USB
EXT
ICSP
www.arduino.cc
RESET
3V3
5V
Gnd
9V
POWER
ANALOG IN
0 1 2 3 4 5

電路圖


3V3
5V
Vin
Power
RST
AREF
Arduino
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
PWM
TX
RX
Digital Input/Output
A0
A1
A2
A3
A4
A5
Analog Input
GND
220 ohm
LED1

程式碼：**Fading.pde**

```
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

void setup()  {
  // declare pin 9 to be an output:
  pinMode(9, OUTPUT);
}

void loop()  {
  // set the brightness of pin 9:
  analogWrite(9, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
```

```
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

編譯這支程式，然後上傳到 Arduino 板子上，過數秒後，就會看到 LED 燈光不斷地改變亮度，一下子漸亮，一下漸暗。

說明：

- L01: brightness 變數用來保存目前的燈光亮度
- L02: fadeAmount 變數用來設定每一次燈光亮度的調整值
- L06: 宣告 pin9 為 output pin, LED 接在 pin9 上
- L11: 使用 analogWrite(9, brightness) 設定 pin9 上的 LED 燈光亮度
- L14: 調整下一次的燈光亮度
- L17~L19: 改變 fadeAmount 燈光亮度調整值，假如 brightness 已達到最亮(255)，就將 fadeAmount 改成 -5，讓燈光下一次的變化改成漸漸變暗，假如 brightness 已達到最暗(0)，就將 fadeAmount 改回 +5，讓燈光下一次的變化改成漸漸變亮。
- L21: 延遲 30ms，這樣肉眼才能看得到 LED 調光的效果

範例照片／影片

Youtube 上正好有段示範利用 PWM 控制 LED 燈光亮度的影片，我們來看看他的示範：

- http://youtu.be/752tMDyvbxE

PWM 原理簡介

數位輸出可以控制訊號的開跟關，開意味著通電，關意味著斷電，如果我們能夠進一步通電的時間比例，就能讓類比輸出產生變化，例如 LED 燈光通電時間為 50%，就可以控制 LED 讓它只有 50% 的亮度，如果把通電時間比例改為 25%，就可以控制 LED 讓它只有 25% 的亮度。這個方法稱為 PWM (Pulse Width Modulation)脈衝寬度調變，PWM 是一個利用數位訊號來控制類比輸出的技術，常用於蜂鳴器、電熱器、馬達或風扇轉速、燈光亮度等的控制。

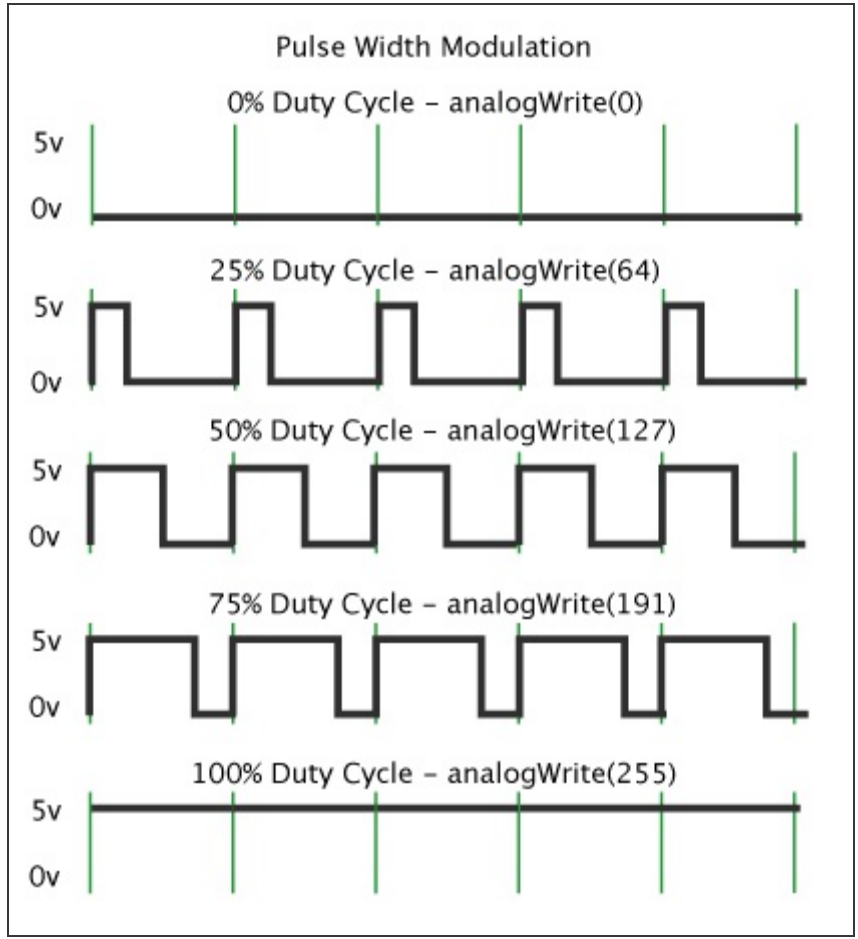
下圖中，垂直的綠線代表一個規律的時間週期，在 Arduino 中，每個週期是 2ms (PWM 頻率為 500Hz)。analogWrite() 的參數值範圍介於 0 到 255 之間，analogWrite(255) 代表產生 100% duty cyle 的輸出(一直通電)，而 analogWrite(127) 則是產生 50% duty cycle 的輸出(一半的時間通電，一半的時間斷電)。

Duty Cycle 為每一個週期通電(on) 的時間比例。

數位輸出可以控制訊號的開跟關，開意味著通電，關意味著斷電，如果我們能夠進一步通電的時間比例，就能讓類比輸出產生變化，例如 LED 燈光通電時間為 50%，就可以控制 LED 讓它只有 50% 的亮度，如果把通電時間比例改為 25%，就可以控制 LED 讓它只有 25% 的亮度。這個方法稱為 PWM (Pulse Width Modulation)脈衝寬度調變，PWM 是一個利用數位訊號來控制類比輸出的技術，常用於蜂鳴器、電熱器、馬達或風扇轉速、燈光亮度等的控制。

下圖中，垂直的綠線代表一個規律的時間週期，在 Arduino 中，每個週期是 2ms (PWM 頻率為 500Hz)。analogWrite() 的參數值範圍介於 0 到 255 之間，analogWrite(255) 代表產生 100% duty cyle 的輸出(一直通電)，而 analogWrite(127) 則是產生 50% duty cycle 的輸出(一半的時間通電，一半的時間斷電)。

Duty Cycle 為每一個週期通電(on) 的時間比例。



你可能會納悶，一直開開關關的 ，LED 會不會閃爍讓眼睛不舒服？答案是不會的，因為開關的頻率很

快，肉眼是看不出 LED 有在閃爍的。

動動腦

1. 如何加快或放慢 LED 亮度變化的速度？(提示：delay() 函式)
2. 改用 pin9 以外的其它 PWM 腳位
3. 多接幾顆 LED，讓每顆 LED 使用不同的速率改變亮度

參考資料

- http://arduino.cc/en/Tutorial/PWM

Arduino 筆記 - Lab4 使用可變電阻調光

實驗目的

使用可變電阻 (potentiometer) 控制 LED 的燈光亮度，達到調光的目的。

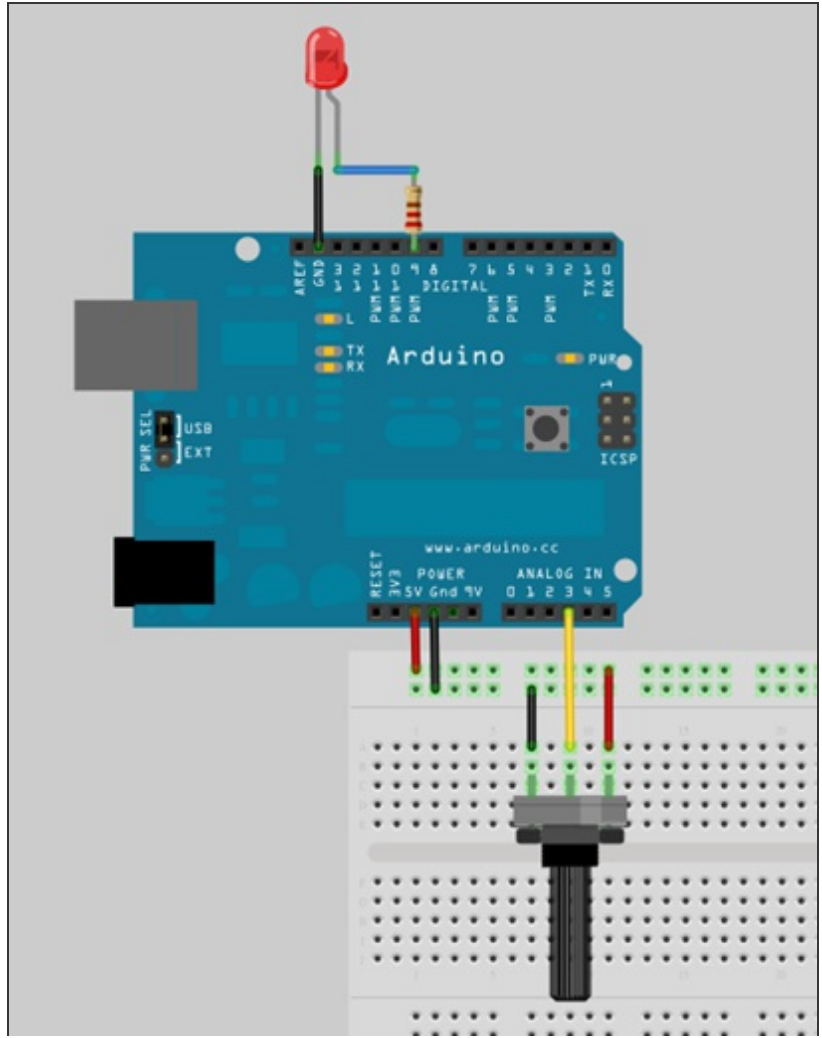
材料

- 麵包板 x 1
- Arduino 主板 x 1
- LED x 1
- 220 ohm 電阻 x 1
- 可變電阻 x 1

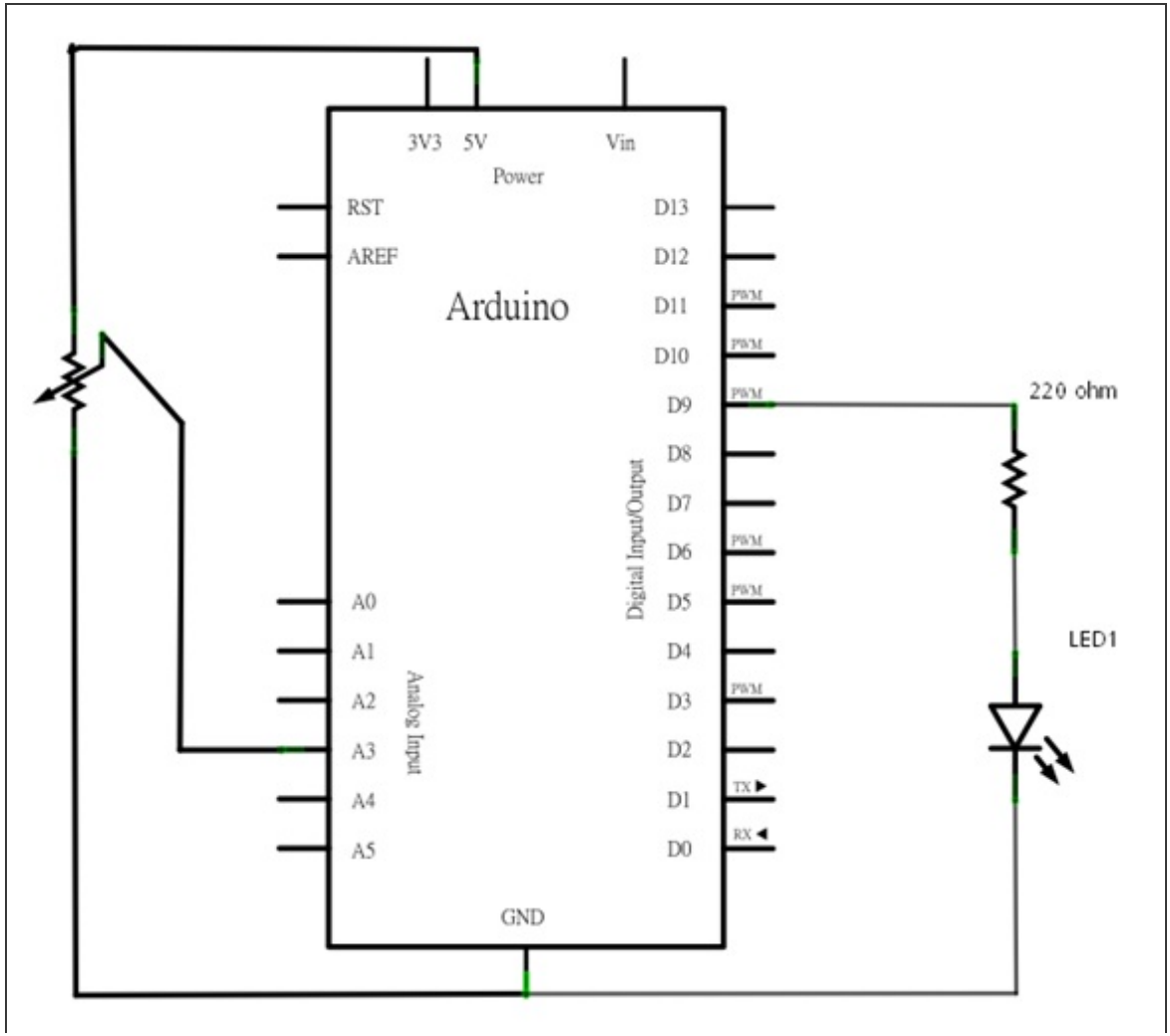
- 單心線 x N

接線

LED 接到 pin9 和 GND，長腳(陽極)串接一顆 220 ohm 電阻到 pin9，短腳(陰極)直接接到 GND 可變電阻中間腳位接到類比輸入(Analog Input) pin3，剩下的兩支腳位，一支接到 5V，另外一支接到 GND


AREF
GND
13 12 11 10 9 8
7 6 5 4 3 2 1 0
DIGITAL
PWM
PWM
PWM
PWM
PWM
PWM
TX
RX
L
TX
RX
Arduino
PWR
PWR SEL
USB
EXT
ICSP
www.arduino.cc
RESET
3V3
5V
Gnd
Gnd
9V
POWER
ANALOG IN
0 1 2 3 4 5

電路圖


3V3
5V
Vin
Power
RST
AREF
Arduino
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
PWM
TX
RX
Digital Input/Output
A0
A1
A2
A3
A4
A5
Analog Input
GND
220 ohm
LED1

程式碼：**potentiometer.pde**

```
int potPin = 3; // select the input pin for the potentiometer
int ledPin = 9; // select the pin for the LED

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(potPin);
  Serial.println(sensorValue, DEC);

  sensorValue = sensorValue/4; // convert from 0-1024 to 0-255
  analogWrite(ledPin, sensorValue);
  delay(150);
}
```

說明：

- L01: 宣告可變電阻所用的類比輸入腳位

- L02: 宣告 LED 燈號腳位
- L05: 設定 SerialPort 的傳輸速率，鮑率為 9600 bps (bit per second)
- L09: 讀取可變電阻讀值並且放到 sensorValue 變數裏
- L10: 使用 Serial.println(sensorValue, DEC) 把電阻值列印到 SerialPort。DEC 代表以十進位顯示數字
- L12: analogRead() 讀進來的是一個 10 位元的數值，值域為 0 到 1023，由於 analogWrite() 的參數只能接受 0 到 255 的數值，所以得將 sensorValue 除以 4，讓 sensorValue 的數值從 0-1023 等比例縮小到 0-255 的範圍。

範例照片/影片

編譯好程式，上傳到 Arduino 後，按下 Serial Monitor 這個按鈕，就會看到 COM Port 不斷收到一串範圍介於 0 到 1023 的數值，這些數值與可變電阻是相關聯的。如果旋轉可變電阻，對應的數值也會跟著改變，而且 LED 燈號的亮度也會跟著改變。

Youtube 上正好有段示範利用可變電阻控制 LED 燈光亮度的影片，我們來看看他的示範：

- Arduino PWM diode dimmer -- http://youtu.be/Ivf-PfZUymo

JavaScript (4) – 在互動網頁中的應用：以功能表為例 (作者：陳鍾誠)

JavaScript 是唯一被各家瀏覽器所共同支援的程式語言，因此在設計網站的時候，我們如果不採用像 Flash

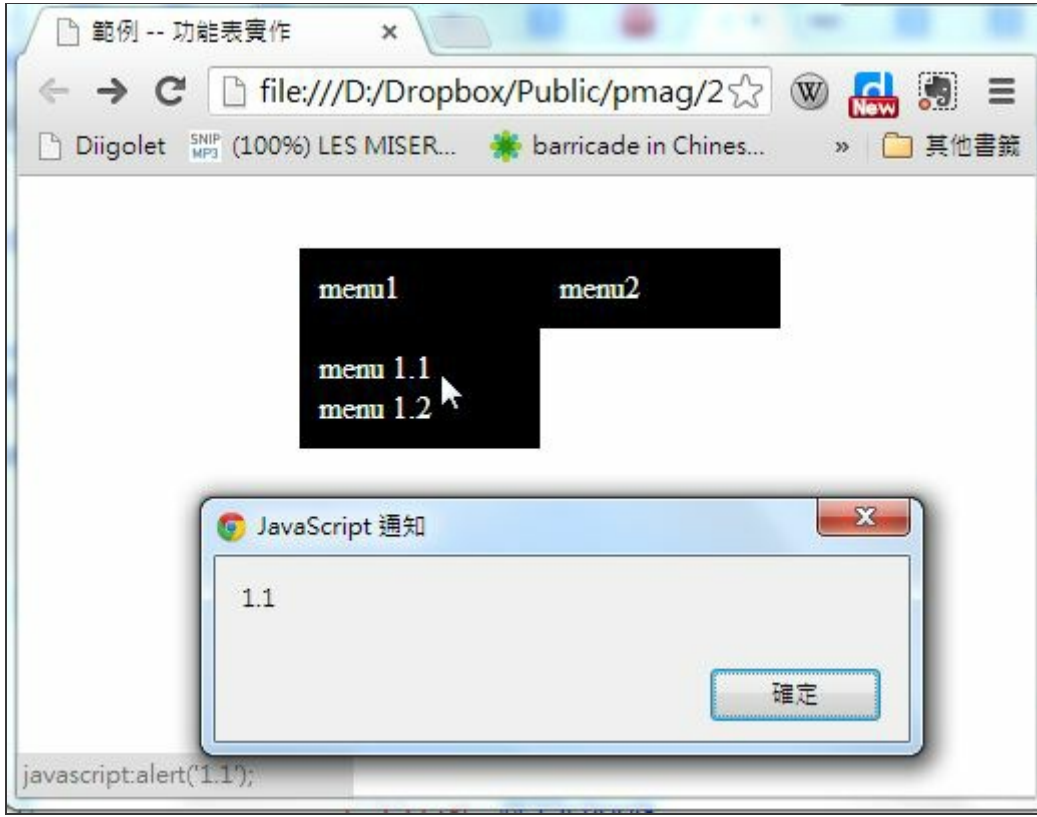
或 Silverlight 這樣的外掛技術，就必須採用 JavaScript 來設計互動式網頁。

在 node.js 這樣的伺服端 JavaScript 開發平台推出之後，我們就能夠採用 JavaScript 同時設計 Client 端與 Server 端的程式，這樣的模式相當的吸引人，我們會在後續的文章中介紹這樣的網站設計方法。

在本文當中，我們將透過 JavaScript 設計一個互動網頁的功能表，以便展示瀏覽器中的 JavaScript 程式是如何運作的。

功能表程式

以下是一個功能表的程式的執行結果，當我們的滑鼠移到功能項上時，就會浮現子功能表，而當我們點下子功能表中的項目時，就會出現一個 alert 視窗，顯示該功能子項被點選的訊息。



功能表程式的執行畫面

以下是這個網頁的原始 HTML 程式碼，其中 <style>...</style> 部分是 CSS 原始碼，而 <script ...

</script> 部分則是 JavaScript 程式碼。

```
<html>
<head>
<title>範例 -- 功能表實作</title>
<style>
.menu   { background-color:black; color:white; padding:10px;
          vertical-align:top; width:100px; list-style-type:none; }
.menu a { color:white; text-decoration:none; }
</style>
<script type="text/javascript">
function show(id) {
  document.getElementById(id).style.visibility='visible';
}

function hide(id) {
  document.getElementById(id).style.visibility='hidden';
}
</script>
  </head>
  <body onload="JavaScript:hide('popup1');hide('popup2');">
```

```
      <ul onmouseover="show('popup1');"  onmouseout="hide('popup1')"
          style="position:absolute; left:100px; top:20px">
        <li id="menu1" class="menu">menu1</li>
        <ul id="popup1" class="menu">
          <li><a href="JavaScript:alert('1.1');">menu 1.1</a></li>
          <li><a href="JavaScript:alert('1.2');">menu 1.2</a></li>
        </ul>
      </ul>
      <ul onmouseover="show('popup2');" onmouseout="hide('popup2')"
          style="position:absolute; left:220px; top:20px">
        <li id="menu2" class="menu">menu2</li>
        <ul id="popup2" class="menu">
          <li><a href="JavaScript:alert('2.1');">menu 2.1</a></li>
          <li><a href="JavaScript:alert('2.2');">menu 2.2</a></li>
          <li><a href="JavaScript:alert('2.3');">menu 2.3</a></li>
        </ul>
      </ul>
  </body>
</html>
```

雖然以上程式只是一個小小的功能表程式碼，但是要能夠讀懂，而且寫得出來，卻要懂相當多的技術才行，這些技術包含 HTML, CSS , JavaScript 與 Document Object Model (DOM)。

程式解析

在 HTML 的一開始，我們用以下語法描述了功能表所需要的 CSS 樣式，當我們套用在像 <li id="menu2" class="menu">menu2 這樣的 HTML 項目上時，就會呈現比較好看的功能表排版格式，而這正是 CSS 樣式的功用。

```
<style>
.menu   { background-color:black; color:white; padding:10px;
          vertical-align:top; width:100px; list-style-type:none; }
.menu a { color:white; text-decoration:none; }
</style>
```

以上的 CSS 語法中，要求功能表要以黑底白字的方式 (background-color:black; color:white;) 顯示，邊緣補上 10 點的空白(padding:10px;)，而且是以向上靠攏 (vertical-align:top;) 的方式，每個功能表的寬度都是 100 點 (width:100px;)，然後不要顯示項目前面的點符號 (list-style-type:none;)。

接著是一段 JavaScript 程式碼的語法，定義了 show(id) 與 hide(id) 這兩個函數，我們可以用這兩個函數在適當的時候讓功能表顯示出來或者是隱藏掉，這樣才能做到「浮現」的功能。

```
<script type="text/javascript">
```

```
function show(id) {
  document.getElementById(id).style.visibility='visible';
}

function hide(id) {
  document.getElementById(id).style.visibility='hidden';
}
</script>
```

然後，開始進入 HTML 的 body 區塊，其中定義了兩組功能表，第一組的內容如下：

```
      <ul onmouseover="show('popup1');"  onmouseout="hide('popup1')"
          style="position:absolute; left:100px; top:20px">
        <li id="menu1" class="menu">menu1</li>
        <ul id="popup1" class="menu">
          <li><a href="JavaScript:alert('1.1');">menu 1.1</a></li>
          <li><a href="JavaScript:alert('1.2');">menu 1.2</a></li>
        </ul>
      </ul>
```

上述區塊最外層的 <ul>... 定義了整個功能表的結構，是由功能母項 <li id="menu1" class="menu">menu1 與子功能表 <ul id="popup1" class="menu">... 所組合而成的，而

<ul onmouseover="show('popup1');" onmouseout="hide('popup1')" style="position:absolute; left:100px; top:20px"> 這一段除了定義了該功能表要顯示在絕對位置 (100,20) 的地方之外，還定義了 onmouseover 與 onmouseout 的事件，這兩個事件讓功能表可以在滑鼠移入時顯示出來，然後在滑鼠移出時隱藏起來，因而做到了浮現式功能表所要求的條件。

由於我們在 <li>... 內的超連結 menu 2.1 使用了 JavaScript 語法，因此在該超連結被點選時，就會有警告視窗顯示 2.1 的訊息，這個訊息僅僅是讓我們知道該功能項被點選了而已，沒有真實的功能。

同樣的、第二個功能表的程式碼也是非常類似的，請讀者看看是否能夠讀者其內容。

```
<ul onmouseover="show('popup2');" onmouseout="hide('popup2')"
    style="position:absolute; left:220px; top:20px">
  <li id="menu2" class="menu">menu2</li>
  <ul id="popup2" class="menu">
    <li><a href="JavaScript:alert('2.1');">menu 2.1</a></li>
    <li><a href="JavaScript:alert('2.2');">menu 2.2</a></li>
    <li><a href="JavaScript:alert('2.3');">menu 2.3</a></li>
  </ul>
</ul>
```

看到這裡，讀者應該大致理解了上述功能表網頁的運作原理，但事實上、我們還漏掉了一行重要的程式

碼，那就是 <body onload="JavaScript:hide('popup1');hide('popup2');"> 這一行，這一行讓浮現功能表能在一開始就處於隱藏狀態，才不會一進來就看到兩個功能表都浮現出來的錯誤情況。

結語

從上述範例中，您可以看到瀏覽器中的 JavaScript ，通常是透過調整網頁某些項目的 CSS 屬性，以達成互動性的功能，這種互動網頁技術，事實上是結合了 HTML+CSS+JavaScript 等技術才能達成的功能，因此這三項技術在瀏覽器當中幾乎是合為一體、可以說是缺一不可的。

在本期中我們說明了互動式網頁的設計方式，但這樣的設計方式非常冗長，對程式人員而言是很大的負擔，因此在互動網頁興起之後，就逐漸出現了各式各樣的互動性 JavaScript 框架，也就是現成的 JavaScript 函式庫，讓我們可以輕易做出很好的互動性，像是 jQuery, ExtJS, YUI, Prototype, Dojo 等互動性函式庫，以便減輕 JavaScript 程式人員的負擔，增加程式員的生產力，在下一期當中，我們將使用最常被使用的 jQuery 框架，再度說明互動網頁的寫法，我們下期見！

R 統計軟體(2)-抽樣與敘述統計 (作者：陳鍾誠)

隨機抽樣

統計的基礎是抽樣，所謂的抽樣就是從母體 (一大群樣本) 當中抽出一些樣本，而在抽樣的時候，我們通常會盡可能的確保樣本的隨機性，以避免抽到的樣本有所偏差。

簡單來說，抽樣是從一群東西(母體) 當中隨機抽取出 x1, x2, ..., xn 等 n 個觀察值的過程，表示如下：

```
母體 => (獨立性) X1, X2, .... , Xn 等 n 個隨機變數相互獨立 => 取出 x1, x2, ...., xn 等
```

在電腦上，我們可以很容易的模擬隨機抽樣，以下是一個使用 R 軟體模擬隨機抽樣的範例，其中指令 sample(1:100, 10) 代表從 1 到 100 的整數當中取出 10 個樣本出來。

```
> x = sample(1:100, 10)
> x
 [1] 12 17 50 33 98 77 39 79  7 26
```

sample 函數的原型是 sample(x, size, replace = FALSE, prob = NULL)，如果 replace 設定為 FALSE，代表已經取過就會被去除，不能重複出現；反之則可以重複出現。

在統計學中，有一些常用的機率模型，都有對應的 R 抽樣函數，以下是一些最常用的機率模型之整理。

機率模型	密度函數	R 函數名稱	說明
二項分布	$\binom{n}{x}p^x(1-p)^{n-x}$	binom(n:size, p:prob)	n:樣本數, p:正面機率, n 次試驗中有 x 個成功的機率

布瓦松分布	$\frac{e^{-\lambda s}\lambda s^x}{x!}$	pois(lambda)	k:期望值, $\lambda = \frac{k}{s}$, 在 s 時間內，事件出現平均 k 次
均勻分布 (Uniform)	$\frac{1}{b-a}$	unif(a:min, b:max)	a:範圍下限, b: 上限 出現機會均等
常態分布 (Normal)	$\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{1}{2}[(x-\mu)/\sigma]^2}$	norm(mean, sd)	中央極限定理：x1+x2+...+xk; 當 k 越大就越接近常態分布
指數分布 (Exponential)	$\frac{1}{b}e^{-x/b}$	exp(rate)	伽瑪分布($a=1, b=\frac{1}{\lambda}$) 布瓦松過程中，第一次事件出現的時間 W

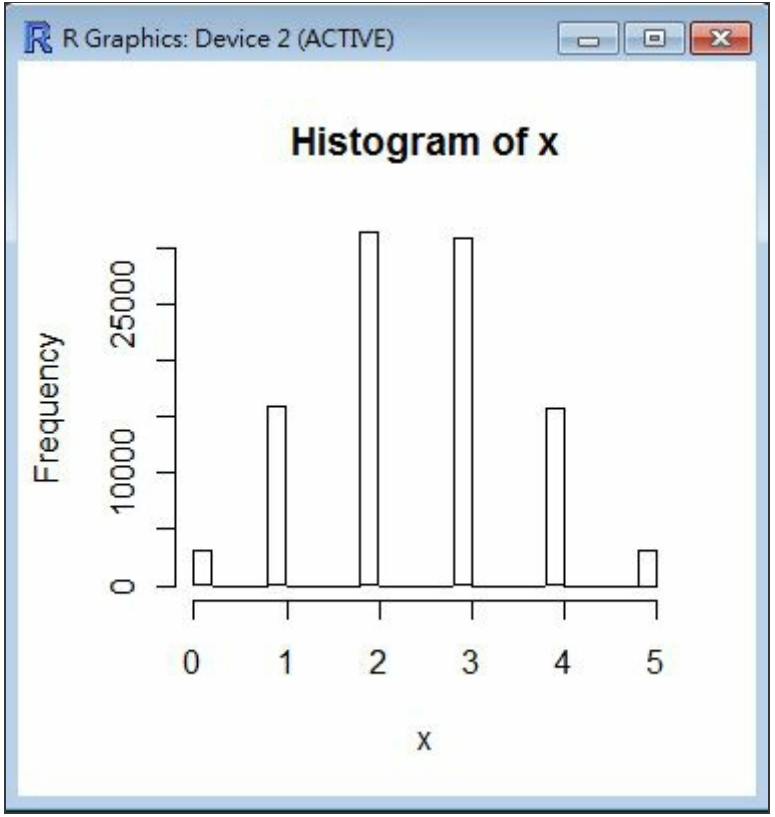
對於每個機率模型，您只要在該函數前若加入 r 這個字，就可以用來產生隨機樣本，以下是一些隨機樣本的產生範例。

```
> rbinom(20, 5, 0.5)
 [1] 4 3 3 4 2 4 3 1 2 3 4 3 2 2 2 4 2 3 1 1
> rpois(20, 3.5)
 [1] 2 1 4 2 1 6 3 6 1 3 3 6 6 0 4 2 6 4 6 2
> runif(20, min = 3, max = 8)
 [1] 3.933526 3.201883 7.592147 5.207603 4.897806 3.848298 4.521461 4.437873
```

```
 [9] 3.655640 5.633540 6.557995 5.430671 6.502675 5.637283 7.713699 5.841052
[17] 6.859493 5.987991 3.752924 7.480678
> rnorm(20, mean = 5.0, sd = 2.0)
 [1] 6.150209 4.743013 3.328734 5.096294 4.922795 6.272768 4.862825 8.036376
 [9] 4.198432 5.467984 2.046450 6.452511 2.088256 5.349187 3.074408 3.628072
[17] 3.421388 7.242598 3.125895 9.865341
> rexp(20, rate=2.0)
 [1] 0.17667426 0.49729383 0.12786107 0.13983412 0.44683515 1.30482842
 [7] 0.28512544 1.61472266 0.23220649 0.39089780 0.05947224 1.42892610
[13] 0.02555552 0.69409186 0.68228242 0.22542362 0.33590791 0.14684937
[19] 0.34995146 0.80595369
```

為了讓讀者能確認這些指令所產生的圖形確實符合分布，讓我們用這些隨機抽樣函數各產生 100,000 個樣本， 然後用 hist() 這個函數繪製統計圖，就能看出這些抽樣函數的效果了，以下是我們的抽樣指令與結果圖形。

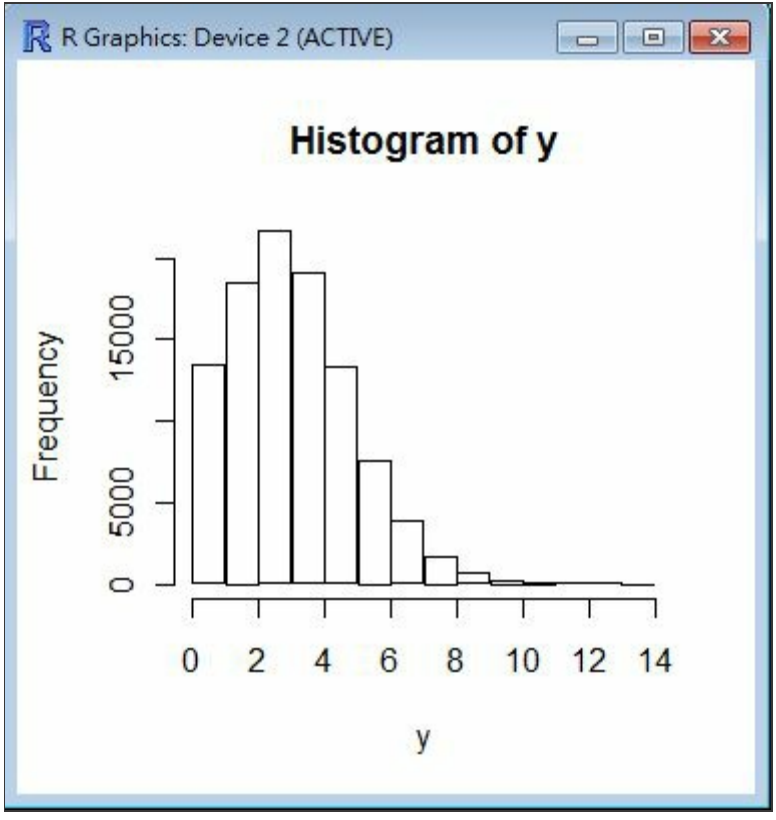
```
> x = rbinom(100000, 5, 0.5)
> hist(x)
```



rbinom(100000, 5, 0.5) 的統計圖

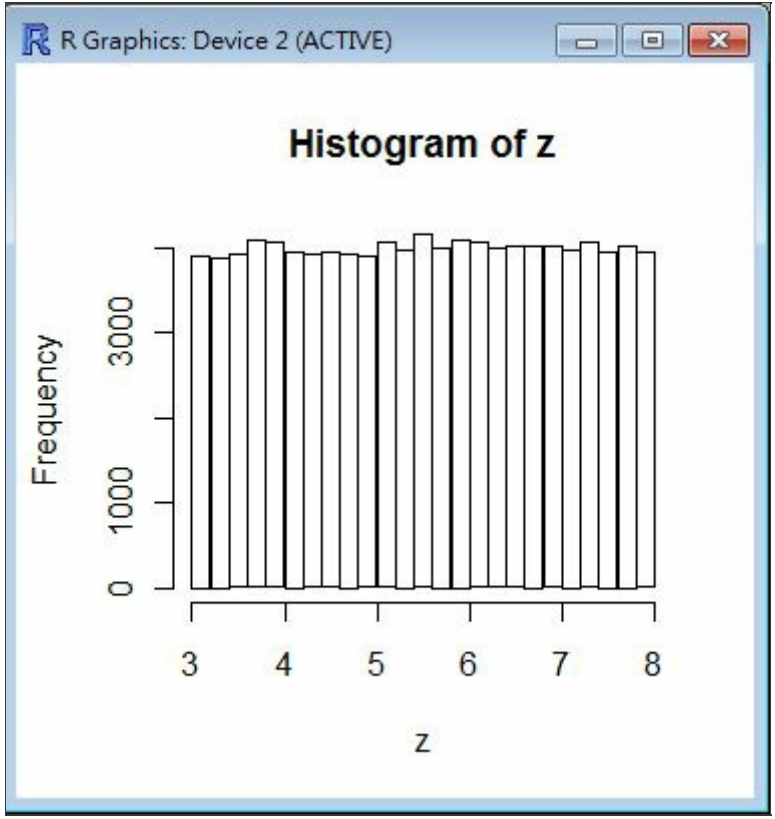
```
> y = rpois(100000, 3.5)
```

```
> hist(y)
```



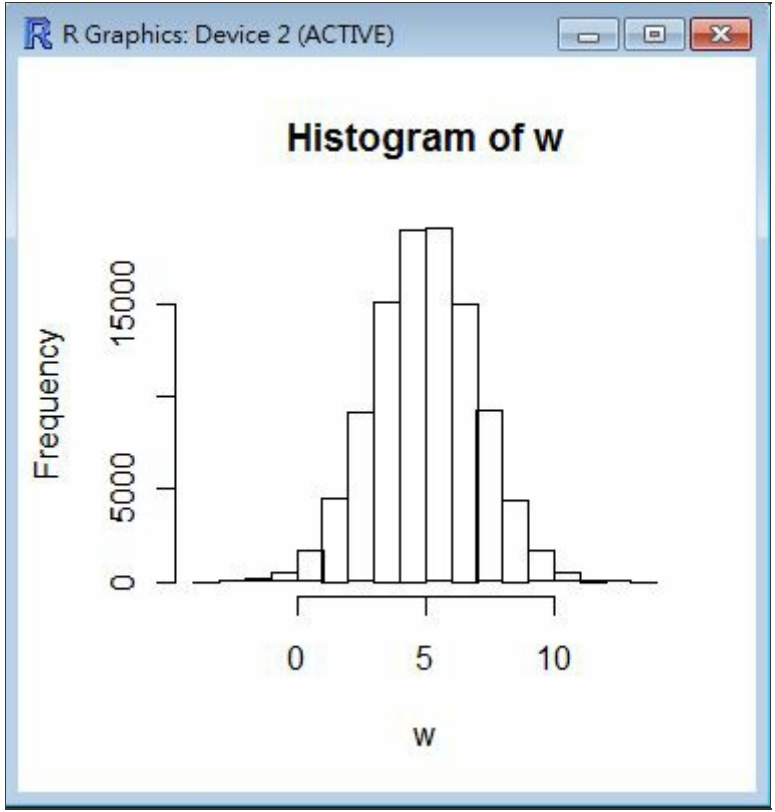
rpois(100000, 3.5) 的統計圖

```
> z = runif(100000, min=3, max=8)
> hist(z)
```



runif(100000, min=3, max=8) 的統計圖

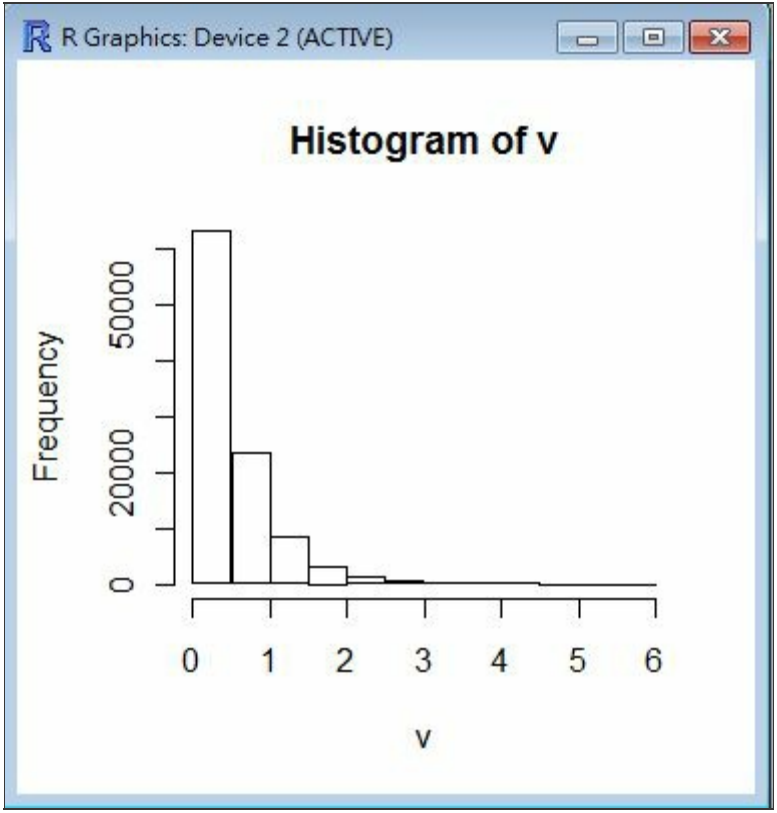
```
> w = rnorm(100000, mean=5.0, sd=2.0)
> hist(w)
```



rnorm(100000, mean=5.0, sd=2.0) 的統計圖

```
> v = rexp(100000, rate=2.0)
```

```
> hist(v)
```



rexp(100000, rate=2.0) 的統計圖

敘述統計：單組樣本的統計

敘述統計乃是隨機抽樣的樣本集合，進行某些計算與繪圖，以便忠實的呈現出樣本的某些特性。這些計算出的數值，以及呈現出來的圖形，可以反映出樣本的某些統計特性，讓統計者能透過數值或圖形，大致了解樣本的統計特徵。

中文名稱	英文名稱	數學公式 / 說明
樣本平均數	Mean	$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$
樣本中位數	Median	樣本排序後最中間位置的數值
樣本變異數	Sample Variance	$S^2 = \sum_{i=1}^{n} \frac{(X_i - \overline{X})^2}{n-1} = \frac{n\sum_{i=1}^{n} X_i^2 - (\sum_{i=1}^{n} Xi)^2}{n(n-1)}$
樣本標準差	Sample Standard Deviation	樣本變異數中的 S 稱為樣本標準差，也就是 $\sqrt{S^2}$
樣本全距	Range	樣本中最大的觀察值減去最小的觀察值 $\omega = X_H - X_L$
離群值	Outlier 或 Wild	離其他樣本很遠，特別大或特別小的樣本值

樣本四分數間距	InterQuartile Range, IQR	第 3 四分位數減掉第 1 四分位數 $IQR = q_3 - q_1$

注意：變異數的定義為 $\delta^2 = E[(X-\mu)^2]$，上述的樣本變異數必須除以 n-1 才是變異數的不偏估計量，而不是除以 n。

R 操作範例：敘述統計

```
> x = sample(1:100, 10)
> x
 [1] 12 17 50 33 98 77 39 79  7 26
> mean(x)
[1] 43.8
> median(x)
[1] 36
> var(x)
[1] 984.1778
> sd(x)
[1] 31.37161
> range(x)
[1]  7 98
> max(x)
```

```
[1] 98
> min(x)
[1] 7
> max(x)-min(x)
[1] 91
> q1 = quantile(x, 0.25)
> q1
  25%
19.25
> q3 = quantile(x, 0.75)
> q3
  75%
70.25
> q3-q1
75%
 51
> iqr(x)
錯誤: 沒有這個函數 "iqr"
> IQR(x)
[1] 51
> fivenum(x)
```

```
[1]  7 17 36 77 98
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   7.00   19.25   36.00   43.80   70.25   98.00
```

敘述統計：繪製統計圖

中文名稱	英文名稱	說明
直方圖	Histogram	根據每個區間的樣本出現次數繪製的長條圖。
肩型圖 (累加分配圖)	Relative Cumulative Frequency Ogive	將累加次數繪製出來的圖形。
莖葉圖	Stem-and-Leaf Diagram	用主幹數字與分支數字表示分布情況的圖形。
盒型圖	Boxplots	由平均值、內籬笆與外籬笆所形成的盒型圖，可看出中心點與離散程度。

說明：盒型圖是由四分位數 q1, q3, 以及內籬笆 f1, f3 (inner fences), 連接值 a1, a3 與外籬笆 F1, F3 (outer fances) 所組成的圖形

- 內籬笆：f1 = q1 - 1.5 iqr; f3 = q3 + 1.5 iqr;

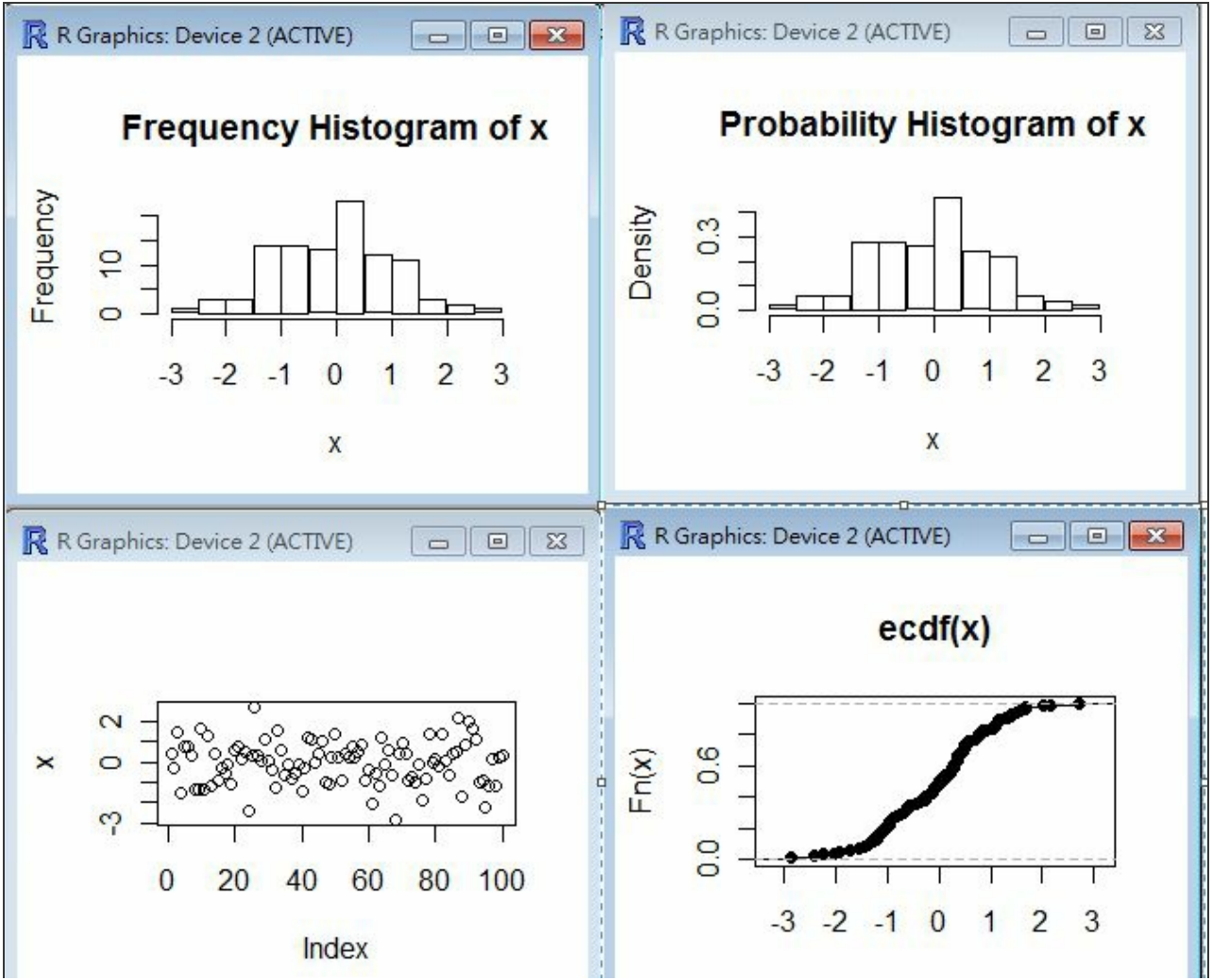
- 外籬笆：F1 = q1 - 3.0 iqr; F3 = q3 + 3.0 iqr;
- 連接值：a1 是大於且最接近 f1 的數據點; a3 小於且最接近 f3 的數據點。

R 操作範例 : 統計圖

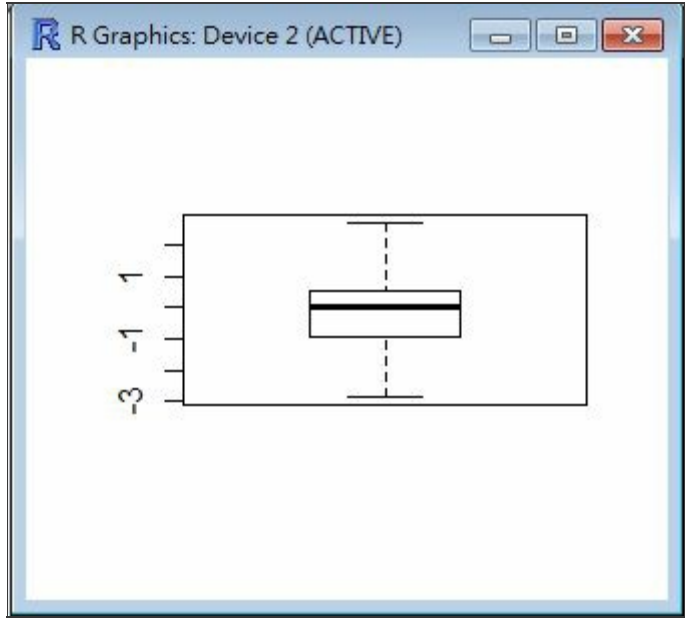
```
> x = rnorm(100)
> x
  [1]  0.389381081 -0.274522826  1.492670583 -1.563228609  0.766405108
  [6]  0.736573742  0.297407135 -1.324130406 -1.376598231  1.661727175
 [11] -1.356590351  1.309122339 -1.193821085  0.365801091 -0.952034088
 [16] -0.277610568 -0.599980091 -0.124105876 -1.107713162  0.560637570
 [21]  0.714449138  0.111969057  0.505171739 -2.418297599  0.318797182
 [26]  2.716646516  0.345289422  0.019434615  1.087758951  0.033917165
 [31] -0.356786424 -1.284809066  1.580411327  0.552931291 -0.615928762
 [36] -0.087069820 -0.814632197 -0.570882510 -0.107731447 -1.453838416
 [41] -0.257115209  1.166866120  1.072692716 -0.022594852  0.441221144
 [46]  1.053900960 -1.025193547 -1.119200587  0.264668203  1.409504515
 [51]  0.241644132 -0.955407800  0.446297381  0.231887649  0.769308731
 [56]  0.269624579  0.496109294  0.822638573 -0.904380789 -0.429527404
 [61] -2.050582772 -0.586973281 -1.192753403  1.158321933 -0.151319360
 [66]  0.558858868 -0.656174351 -2.858964403  0.366785049  0.896958092
 [71]  0.369315063 -0.953560954 -0.762608370 -1.017449547 -0.127738562
```

```
 [76] -1.922030980 -0.839897930  1.332972530 -0.001151104  0.104336360
 [81] -0.208907813  1.401335798  0.019330593 -0.687559289  0.445371885
 [86]  0.504532689  2.168626000 -1.742886230  0.831058071  2.011604088
 [91]  1.676059594  1.132849957 -1.047073217 -0.912548540 -2.235854777
 [96] -1.194104128  0.121106118 -1.178415224  0.214196778  0.280714044
> stem(x)

  The decimal point is at the |

  -2 | 9421
  -1 | 97654433222211000000
  -0 | 998887766664433322111100
   0 | 000111222333333444444455566667788889
   1 | 11112233445677
   2 | 027

> hist(x, main="Frequency Histogram of x")

> hist(x, main="Probability Histogram of x", freq=F)

> Fx = ecdf(x)
```

```
> plot(x)

> plot(Fx)

> boxplot(x)
```


R Graphics: Device 2 (ACTIVE)
Frequency Histogram of x
Frequency
0
10
-3
-2
-1
0
1
2
3
x
R Graphics: Device 2 (ACTIVE)
Probability Histogram of x
Density
0.0
0.3
-3
-2
-1
0
1
2
3
x
R Graphics: Device 2 (ACTIVE)
x
-3
0
2
0
20
40
60
80
100
Index
R Graphics: Device 2 (ACTIVE)
ecdf(x)
Fn(x)
0.0
0.6
-3
-2
-1
0
1
2
3

統計圖



盒狀圖

以上都是對於單組樣本的統計數字與圖形，以下將討論兩組樣本的統計數字

敘述統計：共變異數、兩組樣本的相關度統計

兩組樣本的統計數字，最重要的就是共變異數 (covariance) 相關係數 (correlation) 與了。

共變異數是兩組樣本 X, Y 的樣本與期望值之間差的乘積之期望值，而相關係數則共變異數經過 正規化後的結果，用來表示兩組樣本相關程度，其數值介於 -1.0 到 1.0 之間。

中文名稱	英文名稱	數學公式 / 說明
共變異數	covariance	$\gamma_{XY} = E[(X - E[X])(Y - E[Y])]$
相關係數	correlation	$\phi_{XY} = E[(X - E[X])(Y - E[Y])]/(\sigma_X \sigma_Y)$

讓我們看看 R 軟體中的共變異數函數 cov() 與相關係數 cor() 的操作，如下所示：

```
 runif(10, 1, 5)
> x
 [1] 1.375135 1.863417 2.403693 2.639902 1.694610 4.419406 4.032262 2.147783
 [9] 1.501733 1.497732
> cov(x, x)
[1] 1.144697
> cov(x, x+1)
```

```
[1] 1.144697
> cor(x, x)
[1] 1
> cor(x, x+1)
[1] 1
> cov(x, -x)
[1] -1.144697
> cor(x, -x)
[1] -1
> cor(x, 0.5*x)
[1] 1
> y = runif(10, 1, 5)
> y
 [1] 1.114662 2.358270 2.089179 4.581484 4.170922 2.630044 1.450336 1.320637
 [9] 1.705649 3.506064
> cor(x, y)
[1] -0.04560485
> cor(y, y)
[1] 1
>
```

以上敘述統計所計算出來的數值，都有很直覺的意義，我們就不對這些數值進行詳細的數學說明了，有興趣的讀者請參考統計學的書籍。

在下一期當中，我們將介紹一個重要的機率統計法則：「中央極限定理」，並用 R 軟體進行驗證， 這個法則是理解後續的推論統計程式，像是「信賴區間與檢定的基礎」，我們下期見！

從 C# 看作業系統-(2) 競爭情況、鎖定與生產者/消費者問題 (作者：陳鍾誠)

競爭情況 (Race Condition)

至此，我們已經用 C# 實作了作業系統中的 Thread 與 Deadlock 這兩種概念，但事實上、這兩個概念之間是有關係的，要理解 Thread 與死結之間的關係，就必須從 Race Condition (競爭情況) 這個問題談起。

在多 Thread (或多 CPU) 的情況之下，兩個 thread 可以共用某些變數，但是共用變數可能造成一個嚴重的問題，那就是當兩個 thread 同時修改一個變數時，這種修改會造成變數的值可能錯誤的情況，以下是一個範例。

Thread 1	Thread 2	Thread1+2 (第 1 種情況)	Thread1+2 (第 2 種情況)

counter = 0			
...			
R1 = counter	R1 = counter	T1:R1 = counter	T1:R1 = counter
R1 = R1 + 1	R1 = R1 - 1	T2:R1 = counter	T1:R1 = R1+1
counter = R1	counter = R1	T2:R1 = R1-1	T2:R1 = counter
		T2:counter=R1	T2:R1 = R1-1
		T1:R1 = R1+1	T2:counter = R1
		T1:counter = R1	T1:counter = R1
		...	...
		執行結果 : counter = 0	執行結果 : counter = -1

這種情況並非只在多 CPU 的系統裡會發生，在單 CPU 的多線程系統裡也會發生，因為一個高階語言指令在翻譯成機器碼時，通常會變成很多個指令，這讓修改的動作無法在單一指令內完成，如果這些修改動作

執行到一半的時候，線程被切換了，就會造成上述的競爭情況。

高階語言	組合語言	對應動作
counter ++	LD R1, counter	R1 = counter
	ADD R1, R1, 1	R1 = R1 + 1
	ST R1, counter	counter = R1

這種競爭情況對程式設計者而言是無法接受的，如果程式的執行結果無法確保，那所有的程式都將陷入一團混亂，連 counter++ 這樣的動作都有問題的話，那任何多線程的程式都將無法正確運作。

為了避免這樣的問題產生，一個可能的解決方法是採用鎖定 (lock) 的方式，當我們執行共用變數的修改時，先進行鎖定，讓其他的線程無法同時修改該變數，等到修改完畢後解索後，其他的線程才能修改該變數，這樣就能避免掉競爭情況的問題了。

但是、一但我們能夠進行鎖定的動作，那就可能會造成上述的死結情況，這也正是 Thread 與死結之間的關係，讓我們用一句話總結如下：

> 因為多線程的程式會有競爭情況，為了避免該情況而引入了鎖定機制，
> 但是鎖定機制用得不好就會造成死結。

讓我們先用程式來驗證競爭情況的存在，以下是一個 C# 的競爭情況範例 (當然這種競爭情況是我們刻意造成的)。

檔案：RaceConditon.cs

```
using System;
using System.Threading;
using System.Collections.Generic;

class RaceCondition
{
    static int counter = 0, R1 = 0;
    static void Main(string[] args)
    {
        Thread thread1 = new Thread(inc);
        Thread thread2 = new Thread(dec);
        thread1.Start();
        thread2.Start();
        thread1.Join();
        thread2.Join();
    }
```

```
    static Random random = new Random();

    static void waitAndSee(String msg)
    {
        Thread.Sleep(random.Next(0, 10));
        Console.WriteLine(msg+"     R1:"+R1+" counter:"+counter);
    }

    static void inc()
    {
        R1 = counter;
        waitAndSee("inc:R1 = counter");
        R1 = R1 + 1;
        waitAndSee("inc:R1 = R1 + 1 ");
        counter = R1;
        waitAndSee("inc:counter = R1");
    }

    static void dec()
    {
```

```
            R1 = counter;
            waitAndSee("dec:R1 = counter");
            R1 = R1 - 1;
            waitAndSee("dec:R1 = R1 - 1 ");
            counter = R1;
            waitAndSee("dec:counter = R1");
        }
    }
```

執行結果

```
D:\Dropbox\Public\cs\code>RaceCondition
inc:R1 = counter     R1:0 counter:0
dec:R1 = counter     R1:0 counter:0
inc:R1 = R1 + 1      R1:0 counter:0
inc:counter = R1     R1:0 counter:0
dec:R1 = R1 - 1      R1:0 counter:0
dec:counter = R1     R1:0 counter:0

D:\Dropbox\Public\cs\code>RaceCondition
inc:R1 = counter     R1:0 counter:0
```

```
inc:R1 = R1 + 1      R1:0 counter:0
dec:R1 = counter     R1:0 counter:0
inc:counter = R1     R1:-1 counter:0
dec:R1 = R1 - 1      R1:-1 counter:0
dec:counter = R1     R1:-1 counter:-1
```

要解決以上的競爭情況，必須採用一些協調 (Cooperation) 方法，C# 當中所提供的主要協調方法是 lock 這個語句。簡單來說，C# 的 lock 的實作方式就是採用作業系統教科書中所說的 Monitor 之方法，只是在 lock 的開始加入 Monitor.Enter() 語句，然後在 lock 的結束加入 Monitor.Exit() 語句而已，其方法如下所示。

C# lock	Monitor 語句
lock (_locker) {	Monitor.Enter(_locker);
...	...
critical();	critical();
...	...
}	Monitor.Exit(_locker);

使用 lock 的方式，我們可以很輕易的解決上述程式的競爭情況，以下是該程式加入 lock 機制後的程式碼與執行結果。

檔案：RaceConditonLock.cs

```
using System;
using System.Threading;
using System.Collections.Generic;

class RaceConditionLock
{
    static int counter = 0, R1 = 0;
    static String counterLocker = "counterLocker";

    static void Main(string[] args)
    {
        Thread thread1 = new Thread(inc);
        Thread thread2 = new Thread(dec);
        thread1.Start();
        thread2.Start();
        thread1.Join();
        thread2.Join();
```

```
    }

    static Random random = new Random();

    static void waitAndSee(String msg)
    {
        Thread.Sleep(random.Next(0, 10));
        Console.WriteLine(msg+"     R1:"+R1+" counter:"+counter);
    }

    static void inc()
    {
        lock (counterLocker) {
            R1 = counter;
            waitAndSee("inc:R1 = counter");
            R1 = R1 + 1;
            waitAndSee("inc:R1 = R1 + 1 ");
            counter = R1;
            waitAndSee("inc:counter = R1");
        }
    }
```

```
    static void dec()
    {
        lock (counterLocker) {
            R1 = counter;
            waitAndSee("dec:R1 = counter");
            R1 = R1 - 1;
            waitAndSee("dec:R1 = R1 - 1 ");
            counter = R1;
            waitAndSee("dec:counter = R1");
        }
    }
}
```

執行結果

```
D:\Dropbox\Public\cs\code>csc RaceConditionLock.cs
適用於 Microsoft (R) .NET Framework 4.5 的
Microsoft (R) Visual C# 編譯器版本 4.0.30319.17929
Copyright (C) Microsoft Corporation. 著作權所有，並保留一切權利。
```

```
D:\Dropbox\Public\cs\code>RaceConditionLock
inc:R1 = counter     R1:0 counter:0
inc:R1 = R1 + 1      R1:1 counter:0
inc:counter = R1     R1:1 counter:1
dec:R1 = counter     R1:1 counter:1
dec:R1 = R1 - 1      R1:0 counter:1
dec:counter = R1     R1:0 counter:0

D:\Dropbox\Public\cs\code>RaceConditionLock
inc:R1 = counter     R1:0 counter:0
inc:R1 = R1 + 1      R1:1 counter:0
inc:counter = R1     R1:1 counter:1
dec:R1 = counter     R1:1 counter:1
dec:R1 = R1 - 1      R1:0 counter:1
dec:counter = R1     R1:0 counter:0

D:\Dropbox\Public\cs\code>RaceConditionLock
inc:R1 = counter     R1:0 counter:0
inc:R1 = R1 + 1      R1:1 counter:0
inc:counter = R1     R1:1 counter:1
```

```
dec:R1 = counter     R1:1 counter:1
dec:R1 = R1 - 1      R1:0 counter:1
dec:counter = R1     R1:0 counter:0
```

號誌 **(Semaphore)** 與生產者/消費者問題

```
using System;
using System.Threading;
using System.Collections.Generic;

class ProducerConsumer
{
    static readonly int BUFFER_SIZE = 3;
    static Queue<int> queue = new Queue<int>(BUFFER_SIZE);
    static Semaphore filled = new Semaphore(0, BUFFER_SIZE);
    static Semaphore unfilled = new Semaphore(BUFFER_SIZE, BUFFER_SIZE);
    static Mutex mutex = new Mutex(false);

    static void Main(string[] args)
    {
```

```
        Thread producer = new Thread(new ThreadStart(produce));
        Thread consumer = new Thread(new ThreadStart(consume));
        producer.Start();
        consumer.Start();
        producer.Join();
        consumer.Join();
    }

    static Random random = new Random();

    private static void produce()
    {
        while (true)
        {
            Thread.Sleep(random.Next(0, 500));
            int produceNumber = random.Next(0, 20);
            Console.WriteLine("Produce: {0}", produceNumber);

            unfilled.WaitOne();     // wait(unfilled)
            mutex.WaitOne();                // wait(mutex)
```

```
            queue.Enqueue(produceNumber);

            mutex.ReleaseMutex();           // signal(mutex)
            filled.Release();       // signal(filled)

            if (produceNumber == 0)
                break;
        }
    }

    private static void consume()
    {
        while (true)
        {
            filled.WaitOne();       // wait(filled)
            mutex.WaitOne();                // wait(mutex)
            int number = queue.Dequeue();
            Console.WriteLine("Consume: {0}", number);

            mutex.ReleaseMutex();           // signal(mutex)
            unfilled.Release(); // signal(unfilled)
```

```
            if (number == 0)
                break;
            Thread.Sleep(random.Next(0, 1000));
        }
    }
}
```

執行結果

```
D:\Dropbox\Public\pmag\201304\code>ProducerConsumer
Produce: 7
Consume: 7
Produce: 4
Produce: 12
Consume: 4
Produce: 16
Produce: 1
Consume: 12
Produce: 8
Consume: 16
Produce: 15
```

```
Produce: 7
Consume: 1
Produce: 18
Consume: 8
Produce: 0
Consume: 15
Consume: 7
Consume: 18
Consume: 0
```

另外、在作業系統中有個多行程的經典問題稱為 Dining Philospher (哲學家用餐) 問題，也可以採用 lock 的方法解決，由於這個問題實務上比較不那麼常用，本文中就不再詳細探討此一問題，有興趣的朋友可以看看網路上的解決方法，像是以下這篇 java2s 當中的程式就用 C# 實作解決了此一問題。

- http://www.java2s.com/Tutorial/CSharp/0420__Thread/DiningPhilosopher.htm

在本章中，我們購過透過實作的方式，讓讀者感受作業系統當中的 Thread、Deadlock、Race Condition、與 Semaphore 等概念，希望這樣的說明方式對讀者會有所幫助。

參考文獻

- Threading in C#, Joseph Albahari (超讚！)
  - http://www.albahari.com/threading/
- Agile and Domain-Driven in C#:Multithreaded Producer/Consumer using Semaphore and Mutex in C# 2.0
  - http://agilelover.blogspot.tw/2006/09/multithreaded-producerconsumer-using.html

雜誌訊息

讀者訂閱

程式人雜誌是一個結合「開放原始碼與公益捐款活動」的雜誌，簡稱「開放公益雜誌」。開放公益雜誌本著「讀書做善事、寫書做公益」的精神，我們非常歡迎程式人認養專欄、或者捐出您的網誌，如果您願意成為本雜誌的專欄作家，請加入 程式人雜誌社團 一同共襄盛舉。

我們透過發行這本雜誌，希望讓大家可以讀到想讀的書，學到想學的技術，同時也讓寫作的朋友的作品能產生良好價值 – 那就是讓讀者根據雜誌的價值捐款給慈善團體。 讀雜誌做公益也不需要有壓力，您不需要每讀一本就急著去捐款，您可以讀了十本再捐，或者使用固定的月捐款方式，當成是雜誌訂閱費，或者是季捐款、一年捐一次等都 OK！甚至是單純當個讀者我們也都很歡迎！ 本雜誌每期參考價：NT 50 元，如果您喜歡本雜誌，請將書款捐贈公益團體。例如可捐贈給「羅慧夫顱顏基金會 彰化銀行(009) 帳號：5234-01-41778-800」。(若匯款要加註可用「程式人雜誌」五個字)

想訂閱本雜誌的讀者，請按 雜誌訂閱 連結並填寫表單，我們會在每一期雜誌出刊時寄送通知與下載網址到您的信箱。

投稿須知

給專欄寫作者： 做公益不需要有壓力。如果您願意撰寫專欄，您可以輕鬆的寫，如果當月的稿件出不來，我們會安排其他稿件上場。

給網誌捐贈者： 如果您沒時間寫專欄或投稿，沒關係，只要將您的網誌以 [創作共用的「姓名標示、非商業性、相同方式分享」授權] 並通知我們，我們會自動從中選取需要的文章進行編輯，放入適當的雜誌當中出刊。

給文章投稿者： 程式人雜誌非常歡迎您加入作者的行列，如果您想撰寫任何文章或投稿，請用 markdown 或 LibreOffice 編輯好您的稿件，並於每個月 25 日前投稿到程式人雜誌社團 的檔案區，我們會盡可能將稿件編入隔月1號出版程式人雜誌當中，也歡迎您到社團中與我們一同討論。

如果您要投稿給程式人雜誌，我們最希望的格式是採用 markdown 的格式撰寫，然後將所有檔按壓縮為 zip 上傳到社團檔案區給我們， 如您想學習 markdown 的撰寫出版方式，可以參考 看影片學 markdown 編輯出版流程 一文。

如果您無法採用 markdown 的方式撰寫，也可以直接給我們您的稿件，像是 MS. Word 的 doc 檔或 LibreOffice 的 odt 檔都可以，我們 會將這些稿件改寫為 markdown 之後編入雜誌當中。

參與編輯

您也可以擔任程式人雜誌的編輯，甚至創造一個全新的公益雜誌，我們誠摯的邀請您加入「開放公益出版」的行列，如果您想擔任編輯或創造新雜誌，也歡迎到 程式人雜誌社團 來與我們討論相關事宜。

公益資訊

公益團體	聯絡資訊	服務對象	捐款帳號
財團法人羅慧夫顱顏基金會	http://www.nncf.org/ lynn@nncf.org 02-27190408分機232	顱顏患者 (如唇顎裂、小耳症或其他罕見顱顏缺陷）	銀行：009彰化銀行民生分行 帳號：5234-01-41778-800
社團法人台灣省兒童少年成長協會	http://www.cyga.org/ cyga99@gmail.com 04-23058005	單親、隔代教養.弱勢及一般家庭之兒童青少年	銀行：新光銀行 戶名：台灣省兒童少年成長協會 帳號：103-0912-10-000212-0