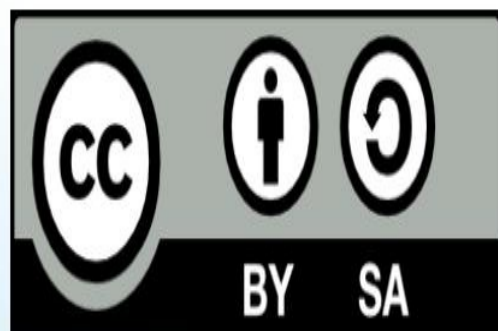


少年科技人

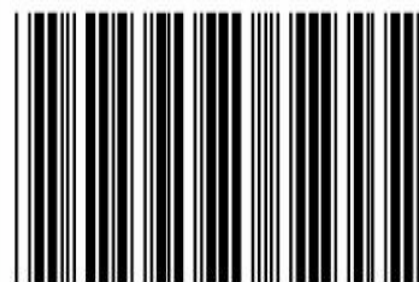
Young Maker 雜誌



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



* 0 1 2 3 7 8 *

愛心條碼

少年科技人雜誌

2014 年 12 月 (第二期)

本期焦點：網頁技術 (Web Technology)

少年科技人雜誌

- 前言
 - 編輯小語
 - 授權聲明
- 本期焦點：網頁技術 (Web Technology)
 - 網路技術的歷史: 從 ARPANet、Internet 到 Web
 - 基於 HTTP/HTML/CSS/JavaScript 的 Web 技術
 - HTML 網頁設計
 - CSS 版面設計
 - JavaScript -- 讓網頁動起來
- 科技人小故事
 - 與大師相遇：Tim Burner Lee
- 影音頻道
 - 電腦網路的概念與歷史 -- Internet 與 Web
- 科學原理
 - 電腦的數學基礎：二進位系統與運算
- 科技人專欄
 - 組成電腦的基礎元件 -- 邏輯閘
- 雜誌訊息
 - 投稿須知
 - 智財權注意事項
 - 參與編輯
 - 公益資訊

前言

編輯小語

本期的焦點是 Web 技術，這是一個以網頁為核心的技術體系，也是目前網路的主力技術。Web 技術主要由 HTML/CSS/JavaScript 這三種技術所組成，其中 HTML 負責網頁的結構，CSS 則是用來讓網頁更美觀的排版語言，JavaScript 則是用來讓網頁具有互動性的程式技術。

當然、除了這三種技術之外，還必須搭配「伺服器端」技術，像是 Ruby on Rail, PHP, ASP.NET, Python Django, 或 Node.js 等在伺服器上執行並輸出網頁的技術，才能讓這些「網頁」技術可以變成完整的「網站」，我們將陸續在未來介紹其中一些伺服器端技術。

另外、延續上期的「電腦的歷史、工業與結構」這個主題，我們在本期開始進入電腦硬體設計的細節，介紹了「二進位系統」與「數位電路邏輯閘」等主題，希望透過這一系列的介紹可以讓讀者快速的理解電腦的設計原理。

----（「少年科技人」雜誌編輯 - 陳鍾誠, 2014/12/01）

授權聲明

本雜誌許多資料修改自維基百科，採用 創作共用：[姓名標示、相同方式分享](#) 授權，若您想要修改本書產生衍生著作時，至少應該遵守下列授權條件：

1. 標示原作者姓名 (包含該文章作者，若有來自維基百科的部份也請一併標示)。
2. 採用 創作共用：[姓名標示、相同方式分享](#) 的方式公開衍生著作。

另外、當本雜誌中有文章或素材並非採用 [姓名標示、相同方式分享](#) 時，將會在該文章或素材後面標示其授權，此時該文章將以該標示的方式授權釋出，請修改者注意這些授權標示，以避免產生侵權糾紛。

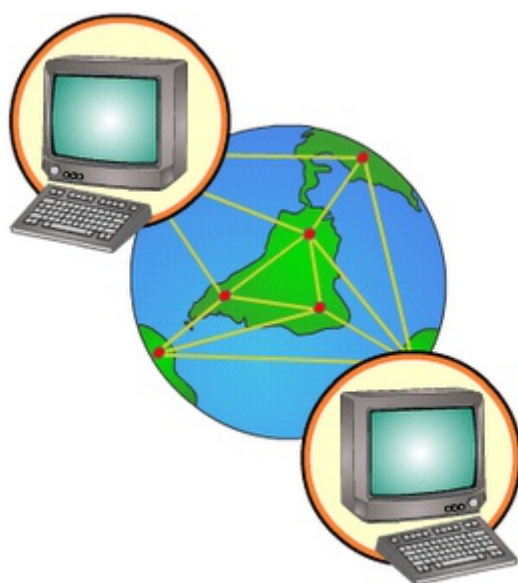
例如有些文章可能不希望被作為「商業性使用」，此時就可能會採用創作共用：[\[姓名標示、非商業性、相同方式分享\]](#) 的授權，此時您就不應當將該文章用於商業用途上。

最後、懇請勿移除公益捐贈的相關描述，以便讓愛心得以持續散播！

本期焦點：網頁技術（Web Technology）

網路技術的歷史：從 ARPANet、Internet 到 Web

在 1950 年代，通信研究者認識到需要允許在不同計算機用戶和通信網路之間進行常規的通信。這促使了分散網路和封包交換的研究。1960 年美國國防部高等研究計劃署（ARPA）出於冷戰考慮建立的 ARPA 網引發了技術進步並使其成為互聯網發展的中心。1973 年 ARPA 網 擴展成可國際互通的互聯網（Internet），第一批接入的有英國和挪威計算機。



圖、Internet

來源：<http://commons.wikimedia.org/wiki/File:Internet.png>

阿帕網（ARPANet）

1960年代、美國國防部先進研究中心（Advanced Research Project Agency, ARPA）是一個超級有錢的研究單位，當時 ARPA 掌控了全美國科學領域研究中70%的資金，雖然 ARPA 所補助的項目常常與國防沒有明顯的關係。

但是、正因為如此，ARPA反而成為許多現代重要科技的發源地，像是網路、電腦繪圖、平行處理、模擬飛行等科技都是在 ARPA 的資助下誕生的。

1966年鮑勃·泰勒（Bob Taylor）成為 ARPA 當中的「資訊處理科技研究室」

（Information Processing Techniques Office, IPTO）的主管，他在任職期間萌發了建構新

型計算機網路的想法，並邀請「阿帕網之父」的拉里·羅伯茨（Larry Roberts）出任信息處理處處長。

1967年，羅伯茨來到高級研究計劃署ARPA，著手籌建「分布式網路」。人員調度和工程設計很順利，不到一年，就提出阿帕網的構想。隨著計劃的不斷改進和完善，羅伯茨在描圖紙上陸續繪制了數以百計的網路連接設計圖，使之結構日益成熟。

1968年，羅伯茨提交研究報告《資源共享的計算機網路》，其中著力闡發的就是讓「阿帕」的電腦達到互相連接，從而使大家分享彼此的研究成果。根據這份報告組建的國防部“高級研究計劃網”，就是著名的“阿帕網”（ARPANet），拉里·羅伯茨也就成為「阿帕網之父」。

1969年底，阿帕網正式投入運行。

最初的「阿帕網」，由西海岸的4個節點構成。第一個節點選在加州大學洛杉磯分校（UCLA），因為羅伯茨過去的麻省理工學院同事L.克萊因羅克教授，正在該校主持網路研究。第二個節點選在斯坦福研究院（SRI），那里有道格拉斯·恩格巴特（D.Engelbart）等一批網路的先驅人物。此外，加州大學聖巴巴拉分校（UCSB）和猶他大學（UTAH）分別被選為三、四節點。這兩所大學都有電腦繪圖研究方面的專家，而泰勒之前的信息處理技術處處長伊凡·澤蘭教授，此時也任教於猶他大學。

但是、除了ARPANet之外，還有像UUCP、Usenet、Bitnet、CSNET和多種商用 X.25 網路等也都在 ARPANet 之後開始發展。

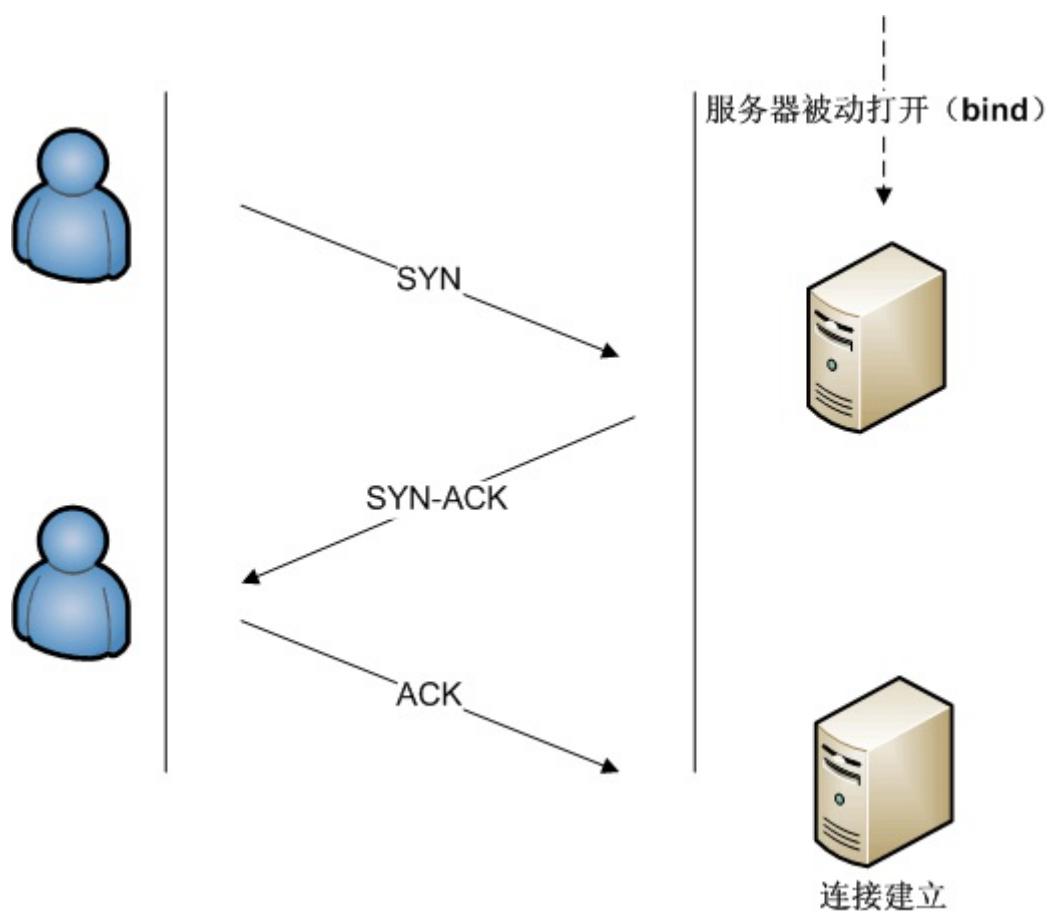
1. 英國 National Physical Laboratory（NPL）所提出採用封包交換式的網路，
2. Michigan Educational Research Information Triad 於 1966 年所開始發展的 Merit Network
3. 法國由 Louis Pouzin 主導設計的研究型網路 於 1973 年開始測試。
4. 1974 年基於 X.25 協議的 SERCnet 網路開始運作，後來成為 JANET。

當時的 ARPANet 由於用了太多專用設備，無法做到和其他網路互通訊息。因此在 1973年文頓·瑟夫（Vinton Cerf）和羅伯特·卡恩（Robert Elliot Kahn）開始思考如何將 ARPA 網和另外兩個已有的網路相連接，尤其是連接衛星網路（SATNET）和基於夏威夷的分組無線業務的ALOHA網（ALOHA NET）瑟夫設想了新的計算機交流協議，最後被稱為傳送控制協議／互聯網協議（TCP/IP）。（他們在2004年也因此獲得圖靈獎）

TCP/IP 協議

1972年，羅伯特·卡恩（Robert E. Kahn）被 DARPA 的信息技術處理辦公室雇用，在那裏他研究衛星數據包網路和地面無線數據包網路，並且意識到能夠在它們之間溝通的價值。在1973年春天，已有的ARPANET網路控制程序（NCP）協議的開發者文頓·瑟夫（Vinton Cerf）加入到卡恩為ARPANET設計下一代協議而開發開放互連模型的工作中。

到了1973年夏天，卡恩和瑟夫很快就開發出了一個基本的改進形式，其中網路協議之間的不同通過使用一個公用互聯網路協議而隱藏起來，並且可靠性由主機保證而不是像ARPANET那樣由網路保證。（瑟夫稱贊Hubert Zimmerman和Louis Pouzin（CYCLADES網路的設計者）在這個設計上發揮了重要影響。）由於網路的作用減少到最小的程度，就有可能將任何網路連接到一起，而不用管它們不同的特點，這樣就解決了卡恩最初的問題。（一個流行的說法提到瑟夫和卡恩工作的最終產品TCP/IP 將在運行“兩個罐子和一根弦”上，實際上它已經用在信鴿上。

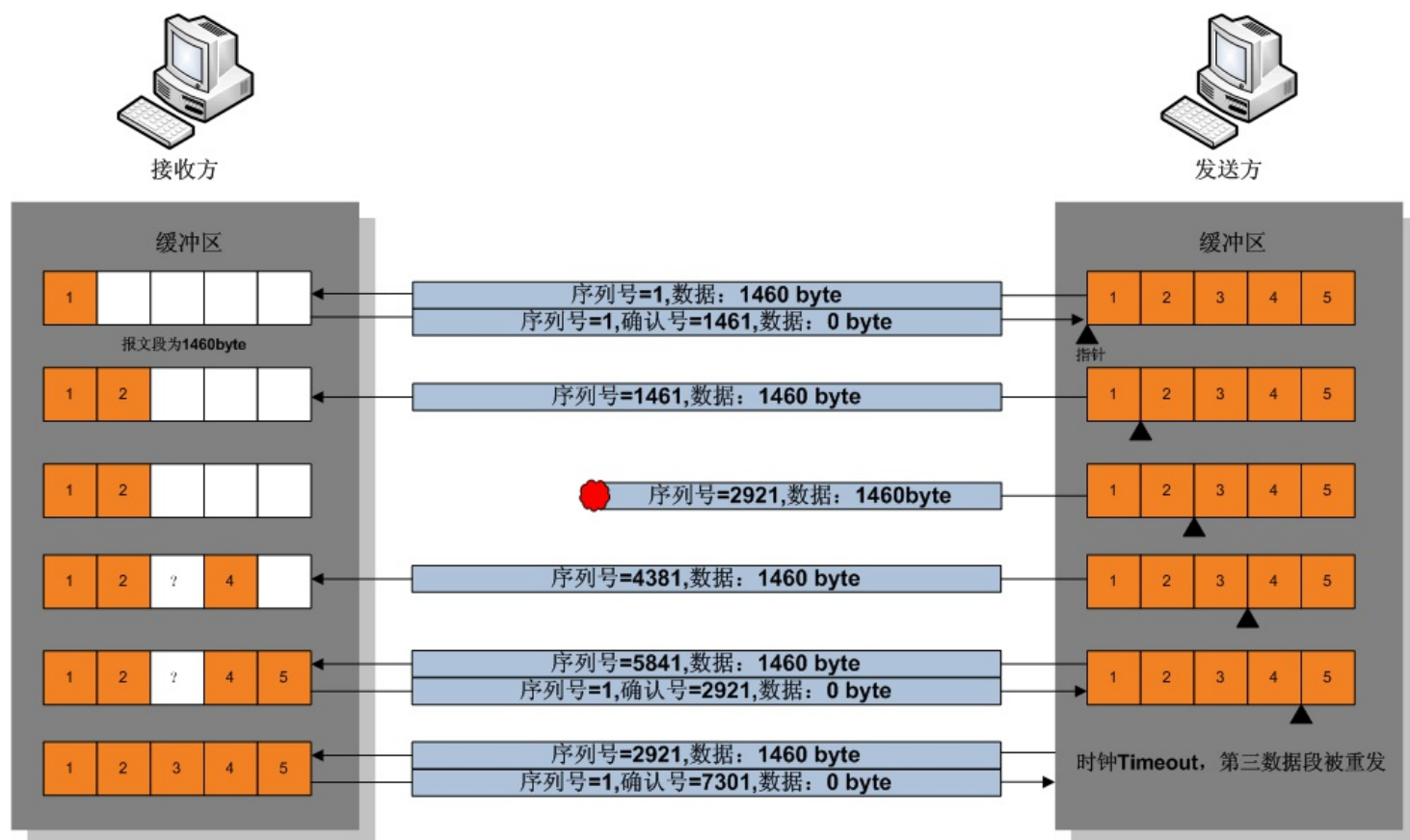


圖、TCP連線的正常建立

來源：

<http://zh.wikipedia.org/wiki/%E4%BC%A0%E8%BE%93%E6%8E%A7%E5%88%B6%E5%8D>

一個稱為網關（後來改為路由器以免與網關混淆）的計算機為每個網路提供一個接口並且在它們之間來回傳輸數據包。這個設計思想更細的形式由瑟夫在斯坦福的網路研究組的 1973年 - 1974年 期間開發出來。（處於同一時期的誕生了PARC通用包協議組的施樂PARC早期網路研究工作也有重要的技術影響；人們在兩者之間搖擺不定。）



圖、TCP資料傳輸協議的運作原理

<http://zh.wikipedia.org/wiki/%E4%BC%A0%E8%BE%93%E6%8E%A7%E5%88%B6%E5%8D>

DARPA於是與BBN、斯坦福和倫敦大學簽署了協議開發不同硬件平臺上協議的運行版本。有四個版本被開發出來——TCP v1、TCP v2、在1978年春天分成TCP v3和IP v3的版本，後來就是穩定的TCP/IP v4——目前因特網仍然使用的標準協議。1975年，兩個網路之間的TCP/IP通信在斯坦福和倫敦大學（UCL）之間進行了測試。

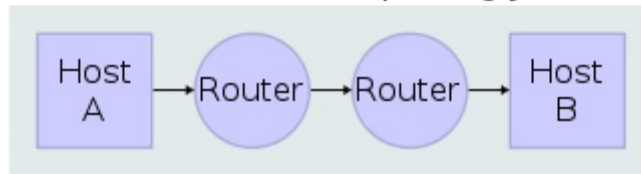
+	Bits 0-3	4-7	8-15	16-31
0	來源連接埠			目的連接埠
32	序列號碼			
64	確認號碼			
96	報頭長度	保留	標誌符	窗口大小
128	檢查碼			緊急指標
160	選用			
160/192+	資料			

- 來源連接埠（16位元長）－辨識傳送連接埠
- 目的連接埠（16位元長）－辨識接收連接埠
- 序列號（32位元長）
 - 如果含有同步化旗標（SYN），則此為最初的序列號；第一個資料位元的序列碼為本序列號加一。
 - 如果沒有同步化旗標（SYN），則此為第一個資料位元的序列碼。

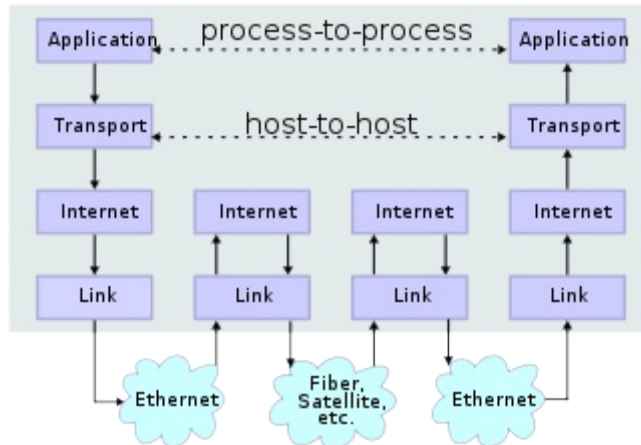
圖、TCP的封包結構

1975年，ARPA網被轉交到美國國防部通信處（Defense Department Communicationg Agence）。此後ARPA網不再是實驗性和獨一無二的了。大量新的網路在1970年代開始出現，包括計算機科學研究網路（CSNET,Computer Science Research Network），加拿大網路（CDnet,Canadian Network），因時網（BITNET,Because It's Time Network）和美國國家自然科學基金網路（NSFnet,National Science Foundation Network）。最後一個網路最終將在它自身被商業網路取代前代替ARPA網作為互聯網的高速鏈路。

Network Topology

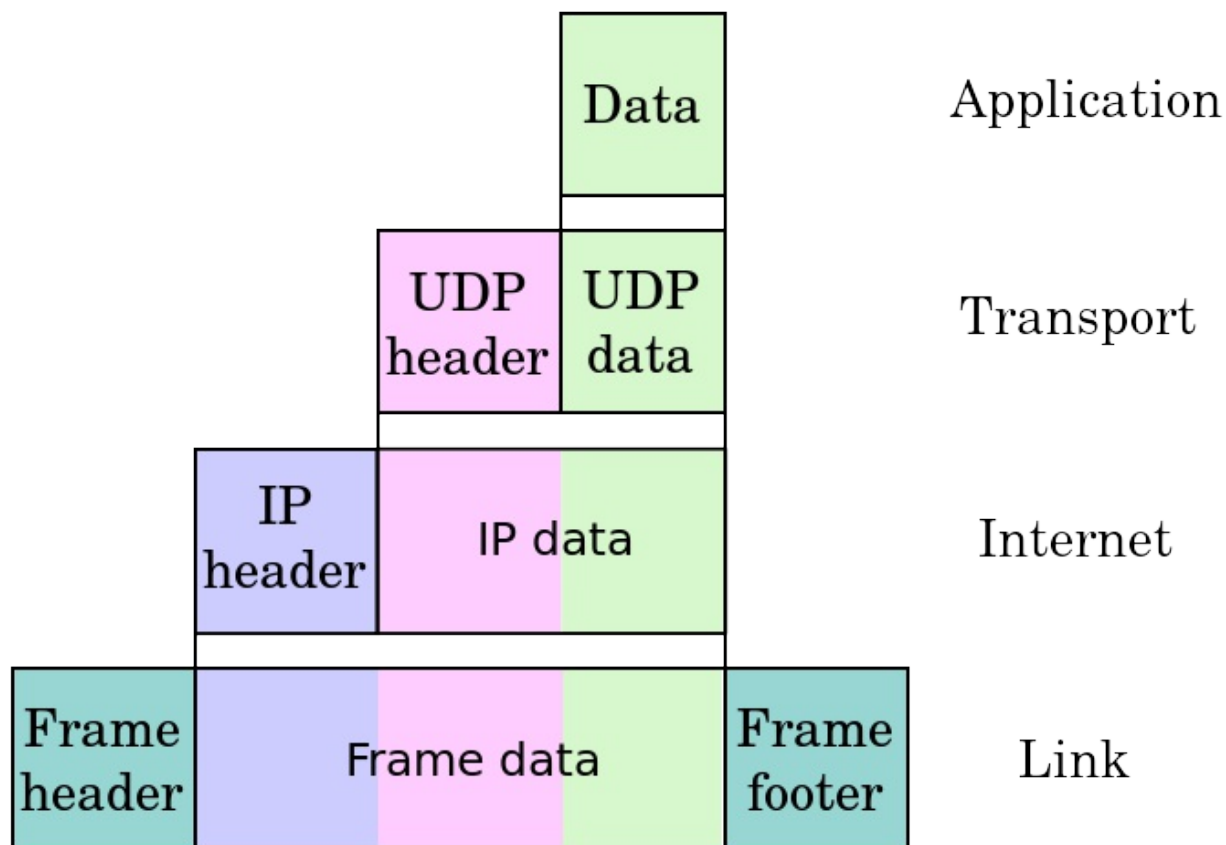


Data Flow



圖、兩個網際網路主機通過兩個路由器和對應的層連接。各主機上的應用通過一些資料通道相互執行讀取操作。

1977年11月，三個網路之間的TCP/IP測試在美國、英國和挪威之間進行。在1978年到1983年間，其他一些TCP/IP原型在多個研究中心之間開發出來。ARPANET完全轉換到TCP/IP在1983年1月1日發生。1984年，美國國防部將TCP/IP作為所有計算機網路的標準。



圖、TCP 封包沿著不同的層應用資料的封裝遞減

1982年中期ARPA網被停用，原先的交流協議NCP被禁用，只允許使用Cern的TCP/IP語言的網站交流。1983年1月1日，NCP成為歷史，TCP/IP開始成為通用協議。

1983年ARPA網被分成兩部分，用於軍事和國防部門的軍事網（MILNET）和用於民間的ARPA網版本。

1985年，因特網架構理事會舉行了一個三天有250家廠商代表參加的關於計算產業使用TCP/IP的工作會議，幫助協議的推廣並且引領它日漸增長的商業應用。2005年9月9日卡恩和瑟夫由於他們對於美國文化做出的卓越貢獻被授予總統自由勳章。

1985年成為TCP/IP協議突破的一年，當時它成為UNIX操作系統的組成部分。最終將它放進了Sun公司的微系統工作站。

1986年，美國國家科學基金會建立了大學之間互聯的骨幹網路NSFnet，這是互聯網歷史上重要的一步。在1994年，NSFNET轉為商業運營。

當免費的在線服務和商業的在線服務興起後，例如Prodigy、FidoNet、Usenet、Gopher等，當NSFNET成為互聯網中樞後，ARPA網的重要性被大大減弱了。系統在1989年被關閉，1990年正式退役。1995年隨著網路開放予商業，網際網路中成功接入的

比較重要的其他網路包括 Usenet、Bitnet 和多種商用 X.25 網路。

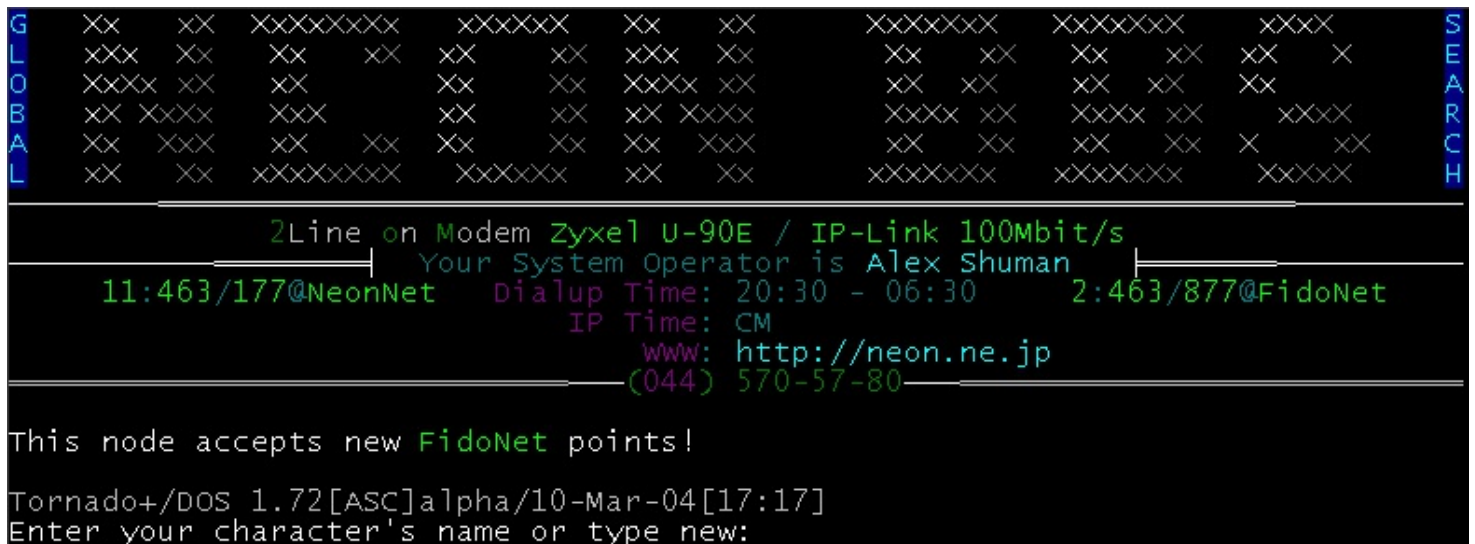
Web 技術的興起



圖、利用Telnet登入香港公共圖書館的資料查詢系統

來源：http://zh.wikipedia.org/wiki/Telnet#mediaviewer/File:HKPL_Telnet.PNG

雖然網路服務已經變多了，但事實上卻還很難進入到家庭裏，因為這種文字型的網路使用介面 (像現在的 PTT, TELENT 那種) 並不容易使用，因此這些網路通常只在研究機構或公司裏被使用著。



圖、Welcome screen of Neon#2 BBS (Tornado)

來源：http://en.wikipedia.org/wiki/Bulletin_board_system#mediaviewer/File:Neon2.png

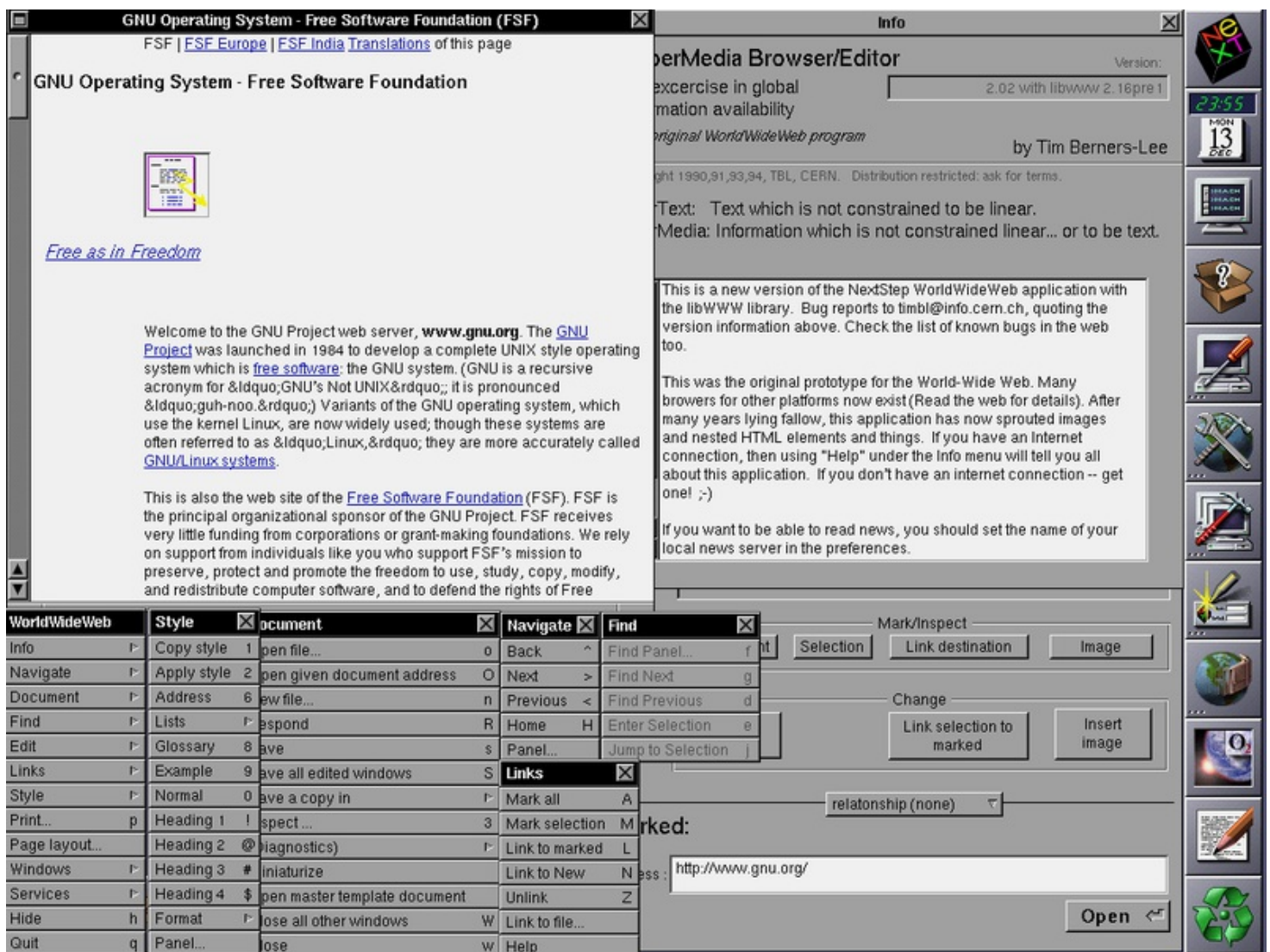


圖、2013年8月30日的進站畫面，探險活寶角色

來源：<http://zh.wikipedia.org/wiki/PTT>

還好、蒂姆·伯納斯-李 (Tim Berners-Lee) 發明了 HTML+URL+HTTP 這三樣法寶，創造了 World Wide Web 這個可以輕易用滑鼠點選就能到處瀏覽的網路架構，讓網路技術開始進入了家庭。

Tim Berners-Lee 是第一個使用 HTML 網頁技術來分享資訊的人。他於 1990年 發明了第一個網頁瀏覽器。在1991年3月，他把這個發明介紹給了他在 CERN (歐洲核子研究組織) 工作的朋友。從那時起，瀏覽器的發展就和網路的發展聯繫在了一起。



圖、1990年，全球第一款網頁瀏覽器—WorldWideWeb於NeXT電腦誕生

在1990年的聖誕假期，Tim Berners-Lee 制作了 Web 所必須的基本工具：那就是第一個 Web 瀏覽器（同時也是編輯器）和第一個網頁伺服器。

Tim Berners-Lee 並不是憑空突然製作出這些工具的，事實上、這是他構思了將近十年之後的結果。

1980年 Tim Berners-Lee 構建的 ENQUIRE 項目。這是一個類似維基百科的超文本在線編輯數據庫。儘管這與我們現在使用的 Web 大不相同，但是它們有許多相同的核心思想。

1989年3月，Tim Berners-Lee 撰寫了《關於信息化管理的建議》一文，文中提及 ENQUIRE並且描述了一個更加精巧的管理模型。1990年11月12日他和羅伯特·卡里奧合作提出了一個更加正式的關於 Web 的建議。在1990年11月13日他在一臺 NeXT 工作站上寫了第一個網頁以實現他文中的想法。

Web 的核心部分是由三個標準構成的：

- 統一資源標識符（URI, 或稱 URL），這是一個統一的為資源定位的系統。
- 超文本傳送協議（HTTP），它負責規定客戶端和服務器怎樣互相交流。
- 超文本標記語言（HTML），作用是定義超文本文檔的結構和格式。

不過、Tim Berners-Lee 的瀏覽器是文字模式的，就像現在你上 PTT 時所看到的類似，因此並不那麼吸引人。

後來在 1993 年時,伊利諾大學厄巴納-香檳分校的 NCSA 組織在1993年所發表了一款視窗界面的瀏覽器,稱為 Mosaic，Mosaic 發表之後讓全世界都瘋狂的下載這個瀏覽器，讓 Web 成為風靡全球的超級網路系統,點燃了網際網路的熱潮。



圖、NCSA Mosaic 3.0執行於Windows

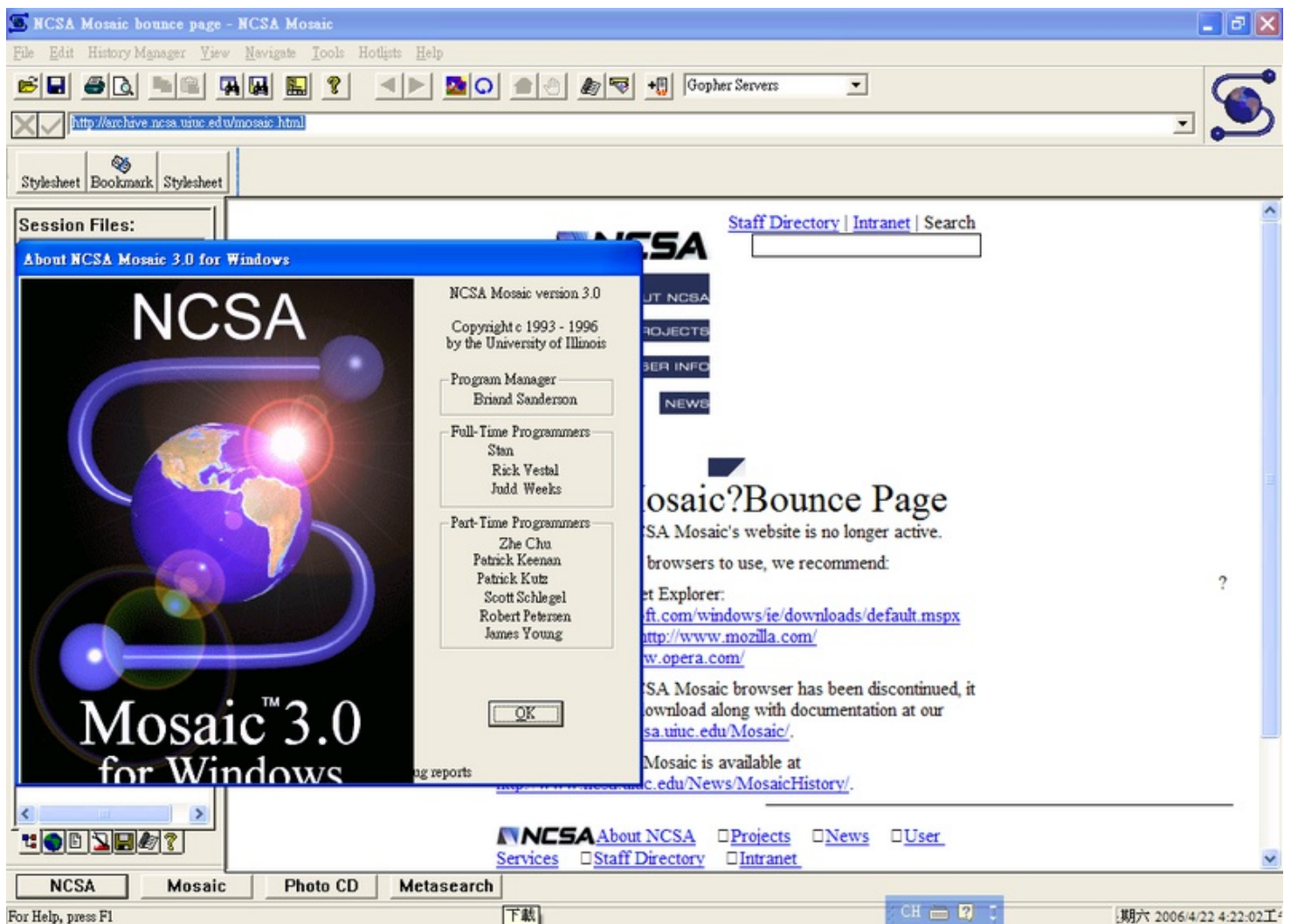
來源：http://zh.wikipedia.org/wiki/File:NCSA_Mosaic.PNG

到底、這款風靡世界的瀏覽器是誰開發出來的呢？這背後又有個傳奇故事。

1992年，當時還是大學生的馬克·安德生（Marc Andreessen）被伊利諾伊大學的電腦應用中心聘為臨時工作人員，馬克·安德生提出設計一種簡單的瀏覽程式想法，能方便的檢索網路資料，於是招聘了幾個程式設計師花了六週的時間開發。

1994年4月，馬克·安德生和矽谷圖形公司的創始人吉姆·克拉克（Jim Clark）在美國加州設立了「Mosaic Communication Corporation」。

Mosaic 的最先版本（1993年1月）只能在 X Window 系統上執行，直到同年9月才支援 Macintosh 和 Windows 等作業系統。



圖、Mosaic v3.0在Windows XP版的截圖，現今的網頁瀏覽器型式大多來自它
來源：<http://zh.wikipedia.org/wiki/Mosaic#mediaviewer/File:Mosaic-v3-screenshot.PNG>

Mosaic 公司成立後，由於伊利諾大學擁有 Mosaic 的商標版權，開發團隊必須徹底重新撰寫瀏覽器程式碼，且瀏覽器名稱更改為 Netscape Navigator，公司名字於1994年11月改名為「網景通訊公司」，此後沿用至今。

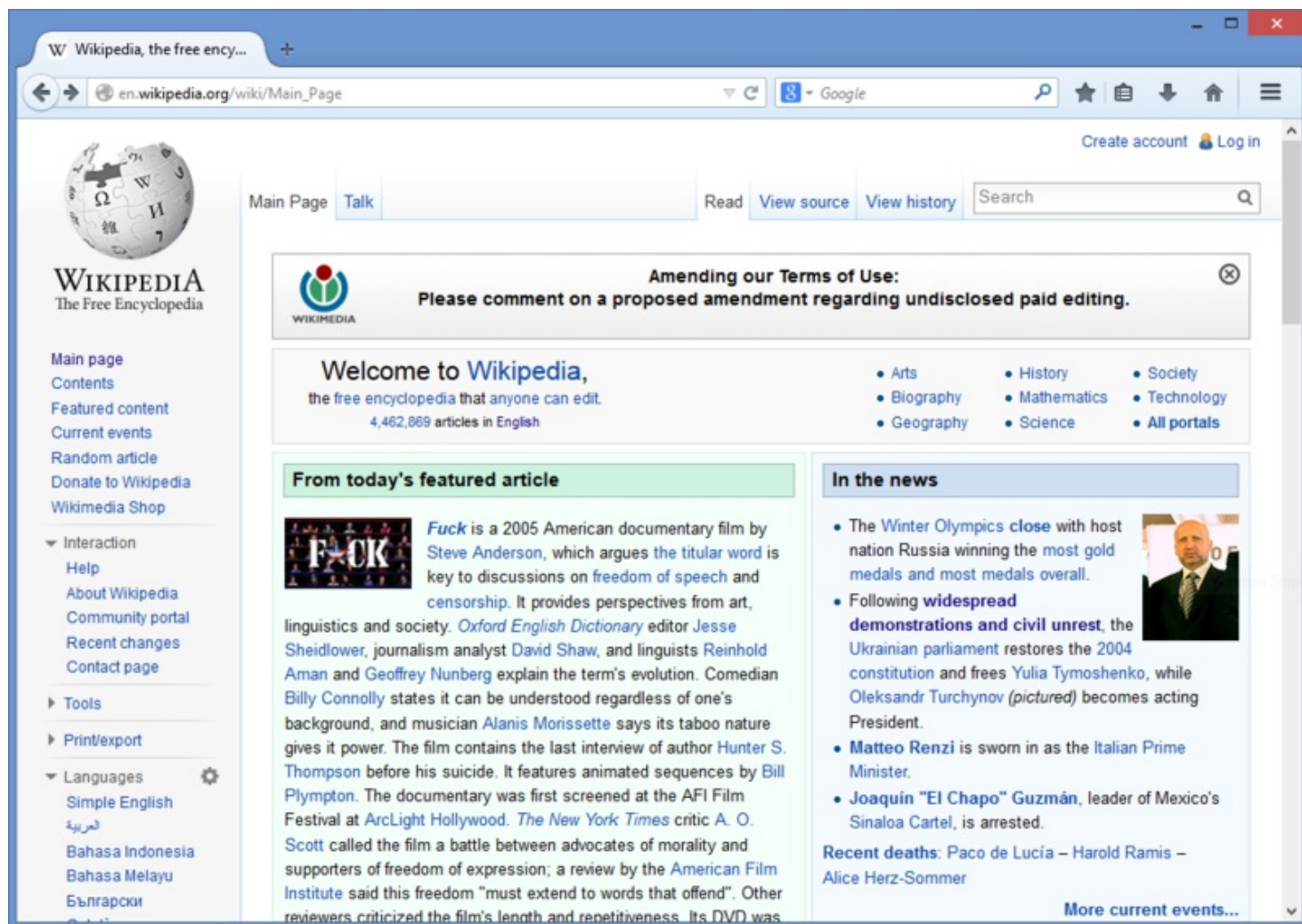
現在 NCSA Mosaic 已無人在更新，無法正確呈現近幾年來新的網頁語法。但由於原始碼早已公開，任何有能力的程式設計師都可以拿它來做進一步的開發，Mosaic 對網際網路的貢獻將永載史冊。

目前主流的瀏覽器，例如：微軟的 Internet Explorer 及 Mozilla Firefox 等，其早期版本皆以 Mosaic 為基礎而開發。微軟隨後買下 Spy Glass 公司的技術開發出 Internet Explorer 瀏覽器，而 Mozilla Firefox 則是網景通訊家開放原始碼後所衍生出的版本。

雖然有了視窗版圖形化的瀏覽器，但是瀏覽器的互動性仍然很受限，像是沒辦法在網頁上呈現動畫或互動式功能表，和現今所使用的瀏覽器功能還差上一截。

為了彌補這個不足，「網景通訊公司」決定在瀏覽器裏加入一個簡單的程式語言，於是就創造出了LiveScript這個程式語言，後來「網景通訊公司」為了與昇陽公司（Sun Inc.）合作之後將其改名為 JavaScript。

Web 發展至此，已經形成了今日 Web 所需要的完整架構，剩下的就是不斷的精進並商業化，然後將全世界的科技人都捲入到 這個網路大發展的浪潮中，像是你看到的 Google、Facebook、雅虎奇摩、淘寶等都是基於 Web 技術的網站。



圖、以 Mozilla Firefox 瀏覽中文維基百科的畫面

現在、不只電腦可以上網，瀏覽 Web，手機和 iPad 等行動裝置也都離不開 Web，一個不能上網的電腦或平板，將會是沒有人想要用的。

這些、都要歸功於那個基於 TCP/IP 與基於 HTTP/HTML/URL/CSS/JavaScript 的 Web 網路架構，以及這段複雜又迷人的網路技術開發史！

參考文獻

- 維基百科: ARPANET

- [維基百科: Internet](#)
- [維基百科: 互聯網歷史](#)
- [維基百科: 傳輸控制協議](#)
- [維基百科: TCP/IP協議族](#)
- [維基百科: Web](#)
- [維基百科: JavaScript](#)
- [維基百科: 網頁瀏覽器](#)
- [維基百科: Mosaic](#)
- [維基百科: NCSA Mosaic](#)
- [維基百科: BBS](#)

【本文由陳鍾誠取材並修改自 [維基百科](#)，採用創作共用的 [姓名標示、相同方式分享](#) 授權】

基於 HTTP/HTML/CSS/JavaScript 的 Web 技術

Web 技術是建構在 Internet 這個基礎網路架構之上的一個應用層網路架構，Internet 網路所採用的協定為 TCP/IP，而 Web 則是在 TCP/IP 之上再加上一層 HTTP 協定。

Network、Internet 與 Web

在英文當中、Network、Internet 與 Web 這些名詞是區分得很清楚的，但是對於中文使用者而言，這三個名詞卻很難釐清，不過對於程式人員而言，區分這幾個名詞是很重要的，特別是 Internet 與 Web 這兩個名詞，首先讓我們來看看這些名詞的差異。

英文中的 Network 泛指任何的網路，包含像公路也叫做 Traffic Network，因此即使在電腦領域，Network 仍然泛指任何網路，包含區域網路、廣域網路、Internet、Ethernet 等等都是 Network。

Internet 是指 1960 年代美國國防部 (Department of Defense, DoD) 所發展出來的那個網路，原本稱為 ARPANET，後來延伸到全世界之後，就稱為 Internet 了。

1990 年 Tim Burner Lee 創造了 HTML、URL、與全世界第一個 WebServer，從此開始創造出一個基於 Internet 的網路架構，這個網路架構被稱為 World Wide Web (簡稱為 Web 或 WWW)，後來在 1992 年當時 Marc Andreessen 招聘了一些程式員開發出一個稱為 Mosaic 的瀏覽器 (Browser, 也就是 Web 的 Client 介面) 之後，推動了 Web 的急速發展。

所以，凡是透過瀏覽器接觸到的網路，通常就是 Web。而不是透過瀏覽器接觸到網路的軟體，則通常是基於 Internet 的應用程式。所以您用 Internet Explorer, Firefox, Chrome, Safari 等瀏覽器所看到的網站，其實都是一個一個的 Web 程式所組成的，而那些不經過瀏覽器，而是要另外安裝的應用程式，像是 Skype、MSN 等，則是 Internet 應用程式。

本書的焦點是 Web 的程式設計，也就是網站的設計，而非像 Skype、MSN 這樣的 Internet 應用程式。

如果您想學習的是 Internet 應用程式的設計，筆者另外有寫一本採用 C# 的程式設計書籍，請參閱該書，該書的網址如下：

- <http://ccckmit.github.io/csharpbook/htm/book.html>

Web 的相關技術

HTTP 是 Web (台灣稱「全球資訊網」、中國大陸稱「萬維網」) 的基礎協定，該協定架構在 TCP/IP 架構之上，屬於應用層的協定。構成 HTTP 的主要兩個應用程式是瀏覽器 (Browser) 與網站 (Web Server)。HTTP 是一個典型的 Client-Server (客戶端-伺服器) 架構的協定，使用者透過 Client 端的瀏覽器連結到 Server 的伺服器，然後由伺服器將結果以 HTML 的網頁格式傳回。HTML 的網頁當中包含了許多超連結 (Hyperlink)，這些超連結連接到某些網址 (URL)，於是使用者可以透過瀏覽器中的超連結，進一步點選其他的網頁，進行網路瀏覽的行為。

絕大部分的使用者，現在都是使用 Internet Explorer, Firefox, Chrome, Safari 等瀏覽器在上網，因此所使用到的就是 Web 程式，本書的焦點也正是這種基於 Web 的程式技術。

Web 相關的技術非常的多，其中最核心的部分是 HTTP/HTML/URL/CSS/JavaScript 等技術，而圍繞著這些核心技術上所發展出來的技術則難以數計，像是 PHP、ASP/ASP.NET、JSP、Ruby on Rail、Python/Dejango、Node.js、XML、JSON、XMLHTTP、AJAX、jQuery 等等，本書後面的章節將採用一個最少語言的架構，以 JavaScript 為核心，採用 HTML + CSS + JavaScript 為主要技術，並搭配 JavaScript 所衍生出的 jQuery 與 Node.js 等技術，以建構出完整的 Web 程式設計概念。

Web = HTTP + HTML + CSS + JavaScript + jQuery + Node.js 所組成的 Web Programming 技術

其中、HTTP 是所有技術的起點，讓我們先來看看 HTTP 到底做了甚麼事。

HTML 與 HTTP

當您用 Browser 看網站的時候，到底 Browser 傳遞什麼訊息給 Server，而 Server 又回傳什麼訊息給 Browser 呢？

粗略的說、一個最簡單的 Web Server 之功能包含下列三個步驟：

- 步驟一：接收瀏覽器所傳來的網址。
- 步驟二：取出相對應的檔案。
- 步驟三：將檔案內容傳回給瀏覽器。

然而、在這個接收與傳回的過程中，所有的資訊都必須遵照固定的格式，規範這個接收/傳送格式的協定，稱為超文字傳送協定 (Hyper Text Transfer Protocol)，簡稱為 HTTP 協定。

HTTP 協定格式的基礎，乃是建構在網址 URL 上的傳輸方式，早期只能用來傳送簡單的 HTML 檔案，後來經擴充後也可以傳送其他類型的檔案，包含影像、動畫、簡報、Word 文件等。

在本文中，我們將先簡介 HTTP 協定的訊息內容，然後在介紹如何以 Java 語言實作 HTTP 協定，以建立一個簡單的 Web Server。

HTTP 協定

當你在 Browser 上打上網址(URL)後，Browser 會傳出一個HTTP訊息給對應的 Web Server，Web Server 再接收到這個訊息後，根據網址取出對應的檔案，並將該檔案以 HTTP 格式的訊息傳回給瀏覽器，以下是這個過程的一個範例。

豬小弟上網，在瀏覽器中打上 `http://ccc.kmit.edu.tw/index.htm`，於是、瀏覽器傳送下列訊息給 `ccc.kmit.edu.tw` 這台電腦。

```
GET /index.htm HTTP/1.0
Accept: image/gif, image/jpeg, application/msword, */*
Accept-Language: zh-tw
User-Agent: Mozilla/4.0
```

```
Content-Length:
Host: ccc.kmit.edu.tw
Cache-Control: max-age=259200
Connection: keep-alive
```

當 ccc.kmit.edu.tw 電腦上的 Web Server 程式收到上述訊息後，會取出指定的路徑 /index.htm，然後根據預設的網頁根目錄 (假設為 c:)，合成一個 c:.htm 的絕對路徑，接著從磁碟機中取出該檔案，並傳回下列訊息給豬小弟的瀏覽器。

```
HTTP/1.0 200 OK
Content-Type: text/html
Content-Length: 438
<html>
    ....
</html>
```

其中第一行 HTTP/1.0 200 OK 代表該網頁被成功傳回，第二行 Content-Type: text/html 代表傳回文件為 HTML 檔案，Content-Length: 438 代表該 HTML 檔案的大小為 438 位元組。

換言之、Browser 會傳遞一個包含網頁路徑的表頭資訊給 Server，而 Server 則會回傳一個 HTML 檔案給 Browser，這樣的傳送格式與規則，就稱為 HTTP 協定。

在 Web 開始成長之後，這樣的模式就顯得有點不足了，因為很多網頁需要使用者填入一些欄位，像是帳號密碼等資訊，但是上述的表頭並沒有欄位資訊預留空間，於是為了回傳這些使用者填入的訊息，有人想到了可以在網址的後面補上參數，這種方式就稱為 GET 的參數傳送方式，以下是一個傳送帳號密碼的表頭訊息，其中的 ?user=ccc&password=1234567 是 Browser 傳送給 Server 的參數。

```
GET /login?user=ccc&password=1234567 HTTP/1.0
Accept: image/gif, image/jpeg, application/msword, */*
Accept-Language: zh-tw
User-Agent: Mozilla/4.0
Content-Length:
```



```
Host: ccc.kmit.edu.tw
Cache-Control: max-age=259200
Connection: keep-alive
```

但是後來，這些填入的欄位越來越多，傳送的參數訊息也越來越長，而這些訊息有些也不希望被顯示在網址列上讓大家看到，因此又發展出了另外一種參數傳遞方式，這種方式就稱為 POST，以下是POST訊息的一個範例。

```
POST /msg.php HTTP/1.0
Accept: image/gif, image/jpeg, application/msword, */*
Accept-Language: zh-tw
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0
Content-Length: 66
Host: ccc.kmit.edu.tw
Cache-Control: max-age=259200
Connection: keep-alive

user=ccc&password=1234&msg=Hello+%21+%0D%0AHow+are+you+%21%0D%0A++
```

上述表頭中的最後一句

「user=ccc&password=1234&msg=Hello+%21+%0D%0AHow+are+you+%21%0D%0A++」是三個欄位參數被編碼後的結果，其中的 user 欄位值是 ccc，password 欄位值是 1234，而 msg 的值則是下列文章。

```
Hello !
How are you !
```

POST 訊息與 GET 訊息不同的地方，除了在 HTTP 的訊息開頭改以 POST 取代 GET 之外，並且多了一個 Content-Length:66 的欄位，該欄位指示了訊息結尾會有 66 個位元組的填表資料，這些資料會被編碼後以文字模式在網路上傳遞。

一旦 Server 取得了 Browser 傳來的 GET 或 POST 訊息後，就可以根據其訊息以進行對應的動作，像是檢查帳號密碼是否正確、將某些訊息存入資料庫、然後在透過 HTML 的方式傳送回應畫面給 Browser，這種透過 HTTP 與 HTML 的簡單互動方式，正是由 World Wide Web 所帶來的核心技術，也讓全世界都捲入了這個影響深遠的網路革命當中。

若您對 WebServer 的運作方式有興趣，可以參考筆者的三個程式與文章，這三個程式分別以 Java、C# 與 JavaScript/Node.js 寫成，這會讓您能夠更深入的體會 WebServer 與 HTTP 的運作原理。

- 如何設計簡單的 WebServer? (Java 版) -- <http://ccckmit.wikidot.com/code:webserver>
- 如何設計簡單的 WebServer? (C# 版) -- <http://cs0.wikidot.com/webserver>
- 如何設計簡單的 WebServer? (JavaScript/Node.js 版) -- <http://ccckmit.wikidot.com/js:nodewebserver>

註：curl 這個工具聽說很適合用來觀察各種協定的運作原理，不過筆者還沒用過，未來會了再寫出來給大家看看！ * <http://curl.haxx.se/download.html>

參考文獻

- How Java Web Servers Work -- http://www.onjava.com/pub/a/onjava/2003/04/23/java_webserver.html
- JDK API : java.net.ServerSocket -- <http://java.sun.com/j2se/1.4.2/docs/api/java/net/ServerSocket.html>
- Hypertext Transfer Protocol (HTTP/1.0) -- <http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html>

【本文由陳鍾誠取材並修改自 [維基百科](#)，採用創作共用的 [姓名標示](#)、[相同方式分享](#) 授權】

HTML 網頁設計

HTML、CSS 與 JavaScript

HTML、CSS 與 JavaScript 是讓 Web Browser 運作的三大技術，要學會 Web 程式設計的第一步，就是要學會這三項技術，在本章中，我們將從最基礎的 HTML 開始。

如果您想要學習這些技術，筆者強烈的推薦您觀看 w3schools 這個網站，網址如下。

- <http://www.w3schools.com/>

學習 HTML、CSS、JavaScript 的順序，最好先從 HTML 開始，因為 HTML 最簡單，而且是整個 Web 技術的表現語言，學會 HTML 之後就可以學習 CSS，看看如何對 HTML 進行格式化套表呈現的動作，然後再進一步學習 JavaScript 這個小型程式語言，以便讓網頁更加動態，增強網頁的的互動功能。

以下是 w3schools 當中這三個主題的網頁，如果您是一個程式設計師，相信您可以用很短的時間就學會這些主題。

- <http://www.w3schools.com/html/default.asp>
- <http://www.w3schools.com/css/default.asp>
- <http://www.w3schools.com/js/default.asp>

一但學會了 HTML、CSS、JavaScript 這三種技術，您就可以開始向 Server 端的技術邁進了。

簡易網頁

HTML 網頁是一種純文字檔案，您只要用記事本這樣的文字編輯器就可以輕易的編出來，以下是一個範例：

```
<html>
<body>
Hello!
</body>
</html>
```

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/hello1.htm>

您可以看到在上述範例中，Hello! 訊息被夾在 <body> 與 </body> 中間，而外層又有 <html> 與 </html> 包覆，像這樣的結構稱為標記語言 (Markup Language)。

如果我們在上述文件中再加上一句 Yahoo! 這樣的標記，那麼網頁中將會出現 Yahoo! 這個詞彙，該詞彙通常是藍色文字，而且可以用滑

鼠點擊，當您點擊之後網頁就會被導到 Yahoo 台灣的首頁，這種點選後會導入另一個網頁的機制是 Web 的最大特色，稱為 Hyperlink (超連結)，而那些被 `<a>` `` 標記起來的文字，就稱為 Hyper Text (超文字)。

```
<html>
<body>
Hello!
<a href="http://tw.yahoo.com/">Yahoo!</a>
</body>
</html>
```

(筆者註：標記 a 當中的 href 是 Hyperlink Reference (超連結引用) 的縮寫，而 a 則是 anchor (錨) 的意思)。

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/hello2.htm>

而這也正是 HTML 之全名 Hyper Text Markup Language 的意義，也就是一種含有超連結的標記語言。

由於 HTML 標記語言是有層次性的，因此在排版時我們經常都會使用縮排以凸顯出這種層次感，例如以上的範例經過縮排之後就變成了如下的情況。

```
<html>
  <body>
    Hello!
    <a href="http://tw.yahoo.com/">Yahoo!</a>
  </body>
</html>
```

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/hello3.htm>

這樣的結構看起來更加一目了然，能夠凸顯出每個層次之間的包含關係。

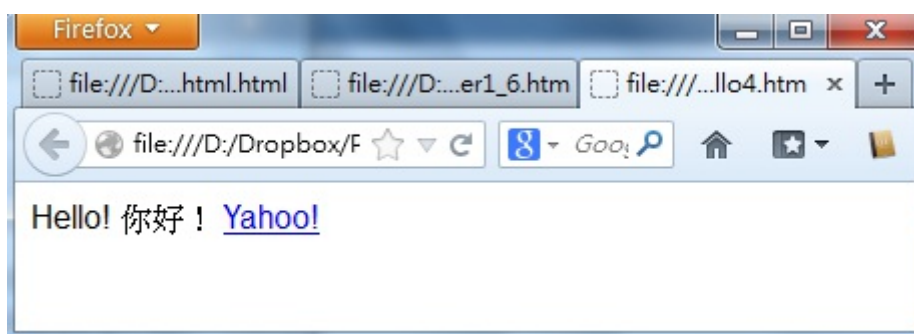
HTML 的表頭

雖然上述範例已經是 HTML 網頁了，而且也通常可以正常檢視了，但是還不夠完

整，一個完整的 HTML 網頁通常要包含足夠的表頭資訊，以便讓瀏覽器能正確的辨認這個網頁的類型與編碼方式，以下是一個範例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset='utf-8' >
  </head>
  <body>
    Hello! 你好!
    <a href="http://tw.yahoo.com/">Yahoo!</a>
  </body>
</html>
```

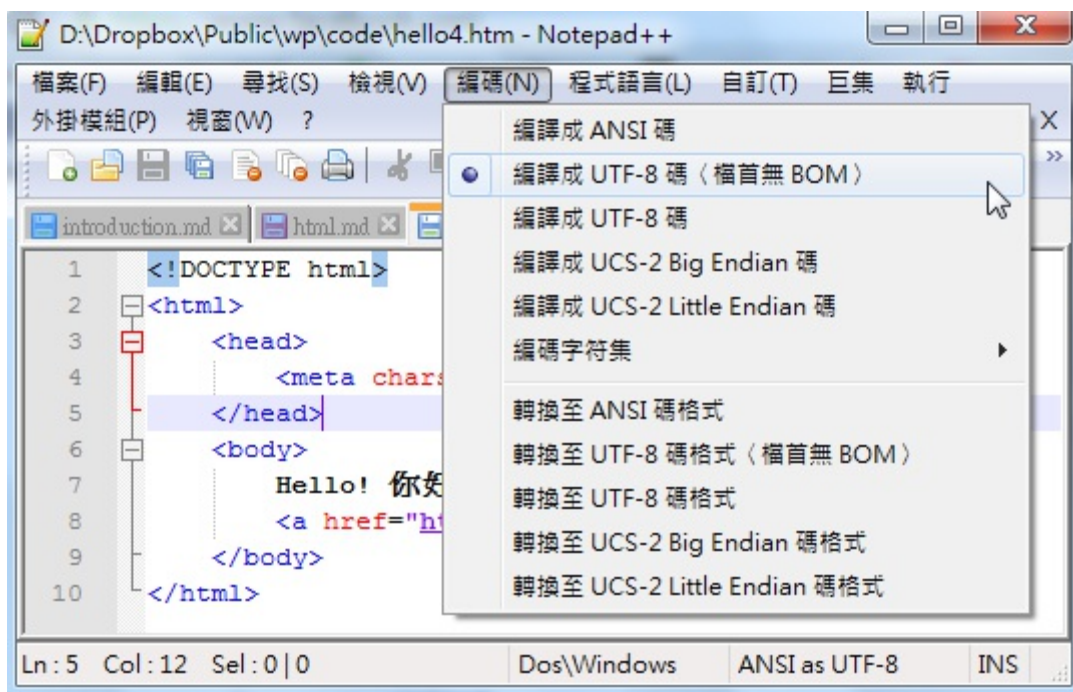
檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/hello4.htm>



圖、上述網頁的檢視畫面

其中第一句的 <!DOCTYPE html> 是告訴瀏覽器，這個文件是一個 HTML 檔案，而 <meta charset='utf-8'> 這句話，則是告訴瀏覽器這個文件的編碼是用 UTF-8 的格式。

所以您必須將上述檔案，儲存成 unicode 的 UTF-8 格式，這樣網頁才能正確運作，否則就有可能出現亂碼的情況，以下是筆者使用 Notepad++ 設定 UTF-8 格式的情況。



圖、筆者使用 Notepad++ 設定 UTF-8 格式的畫面

格式化

從上述範例中，您可以看到即使我們在 HTML 的原始文字檔案中進行換行，但是呈現出來的結果卻沒有換行，如果您真的想要強制換行的話，在 HTML 當中必須用 `
` 這個標記，才能讓文字真正換行。

但是強制換行並不是一個好的寫法，比較好的方法是用段落標記 `<p>...</p>` 把您的段落包起來，這樣在段落結束後就會換行了。

另外、還有 `...` 標記 (也可以用 ` ... `) 可以讓文字變粗體，而 `<i>...</i>` 指令則可以讓文字呈現斜體，` ... ` 標記可以強調文字，而 `<code>...</code>` 可以讓文字以類似程式碼的字型輸出。

若要讓文字變成上標，可以用 `^{...}`，或用 `_{...}` 則可以讓文字變下標，以下範例綜合的呈現了這些格式化標記的結果。

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset='utf-8'>
  </head>
```



```
<body>
```

```
一般字體 (Plain Text) <BR/>
```

```
<i>斜體 (Italic)</i> <BR/>
```

```
<b>粗體 (Bold)</b> <BR/>
```

```
<p>
```

```
<strong>明顯 (Strong)</strong>
```

```
<em>強調 (Emphasized)</em>
```

```
</p>
```

```
<p>
```

```
這是 <sup>上標</sup> 與 <sub>下標</sub> 的顯示情
```

```
況!
```

```
</p>
```

```
</body>
```

```
</html>
```

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/format.htm>



圖、各種格式標記的網頁檢視畫面

標題

HTML 的章節標題共有六種層次，從 h1 到 h6，通常 h1 代表最大的第一層標題，會用最大的字型顯示，而 h6 則是最小的第六層標題，會用最小的字型顯示。


```
<h1>第 1 層段落標題</h1>  
<h2>第 2 層段落標題</h2>  
<h3>第 3 層段落標題</h3>  
<h4>第 4 層段落標題</h4>  
<h5>第 5 層段落標題</h5>  
<h6>第 6 層段落標題</h6>
```

而整個網頁的標題則是用 <title>...</title> 進行標記，如下所示。

```
<title>網頁標題</title>
```

以下是這些標題的使用範例，您可以點選看看。

檢視檔案：https://dl.dropbox.com/u/101584453/web/wp/code/header1_6.htm



圖、Header 1-6 的網頁檢視畫面

影像 Image

在 HTML 中，要顯示影像或圖片，只要使用 就行了。如果要指定大小，就可以加上 width 與 height 屬性，如以下範例所示。

```



```

上述第一個指令 `` 會按照原圖大小顯示，而 `` 則會將圖形縮放到寬度 200，高度 300 的大小，最後一個用 20% 代表縮放到寬度為瀏覽器畫面視窗的百分之二十那麼大。

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/img.htm>



圖、img 指令的網頁檢視畫面

評論：竟然趁機為自己的書打廣告，真是太惡劣了！....

表格 Table

在 HTML 當中、有很多排版都是使用表格的方式製作的，例如「側欄、上方橫幅」等等，表格的語法如以下範例所示。

```
<table>
<tr>
```

```
<th></th>
<th>欄 1</th>
<th>欄 2</th>
</tr>
<tr>
<th>列 1</th>
<td>1, 1</td>
<td>1, 2</td>
</tr>
<tr>
<th>列 2</th>
<td>2, 1</td>
<td>2, 2</td>
</tr>
</table>
```

表格的語法都是由 <table>...</table> 所夾住的，其中 <tr>...</tr> 會夾住一個列，而每個列又可以分成數個欄，這些欄可能用 <th>..</th> 或 <td>...</td> 夾住，其中 th 通常是用在標頭、而 td 則是用在內容。

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/table.htm>



The screenshot shows a web browser window with the title 'table.h x'. The address bar shows 'Diigolet' and a folder icon labeled '其他書籤'. The main content area displays a table with the following structure:

	欄 1	欄 2
列 1	1, 1	1, 2
列 2	2, 1	2, 2

表格範例的顯示結果

項目 list

HTML 的項目是用 `...` 語法所框住的，如果在項目的外面加上 `...`，那就會顯示無編號的項目清單，若加上 `...`，那就會顯示有編號的項目清單。

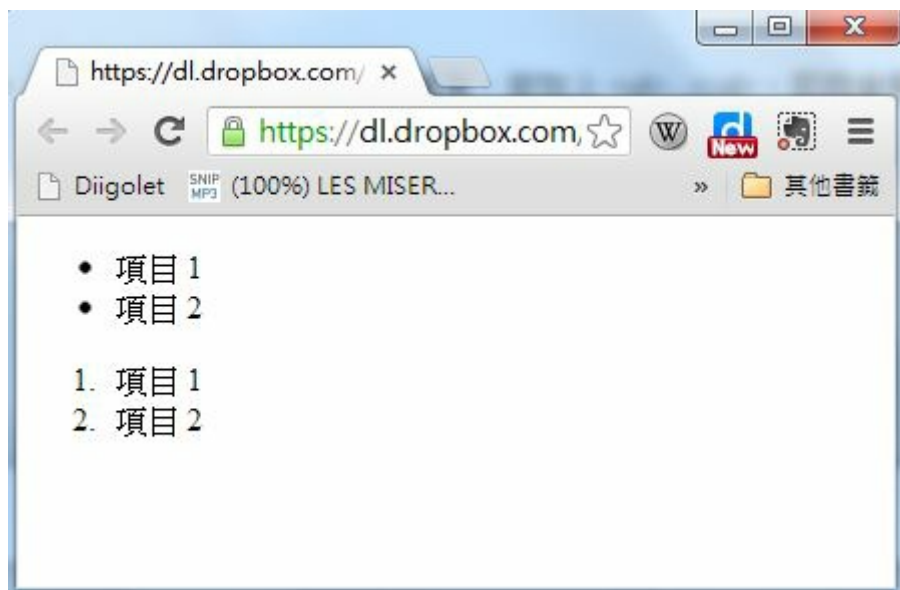
無編號項目清單：

```
<ul>
<li>項目 1</li>
<li>項目 2</li>
</ul>
```

有編號項目清單：

```
<ol>
<li>項目 1</li>
<li>項目 2</li>
</ol>
```

檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/list.htm>



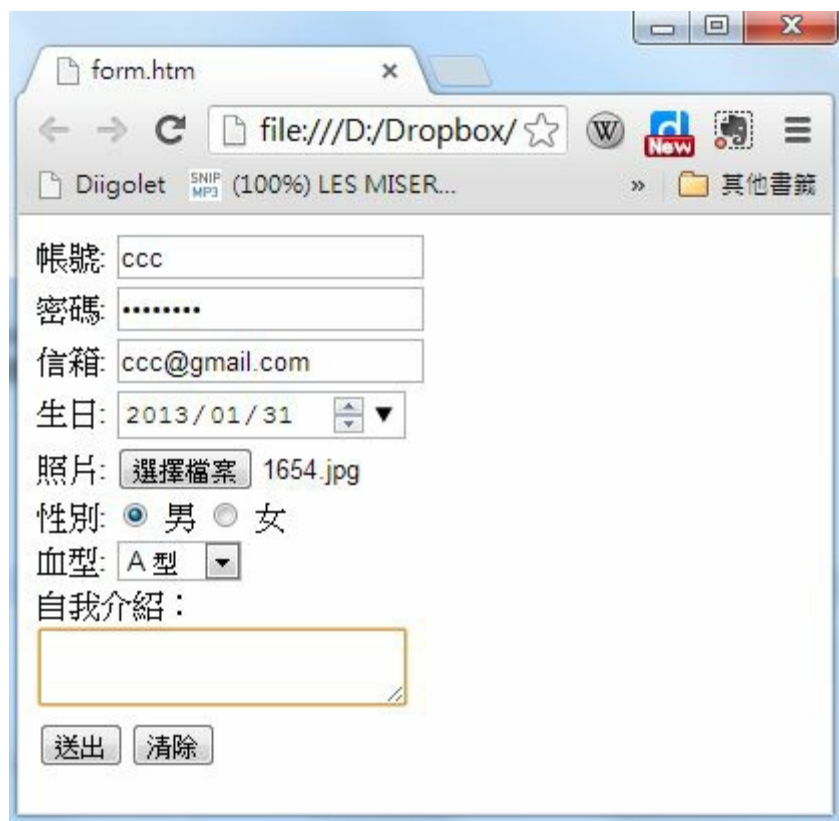
項目清單的顯示結果

表單 Form

在 HTML 當中，表單 (Form) 是指可以讓使用者進行輸入的元件，其語法是用 <form> ...</form> 夾住一堆的輸入元件，這些輸入元件包含 input (輸入)、textarea (文字區)、select (選項) 等，其中的 input 還可以根據其 type 欄位顯示成 checkbox, color, date, datetime, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, url, week 等各種不同的輸入欄形式，以下是一個表單的範例。

```
<form action="signup" method="post">
帳號: <input type="text" name="user"/><br/>
密碼: <input type="password" name="password"/><br/>
信箱: <input type="email" name="email"/><br/>
生日: <input type="date" name="birthday"/><br/>
照片: <input type="file" name="picture"/><br/>
性別: <input type="radio" name="sex" value="male" checked/>
  男
      <input type="radio" name="sex" value="female"/> 女<br/>
血型: <select name="BloodType">
      <option value="A">A 型</option>
      <option value="B">B 型</option>
      <option value="AB">AB 型</option>
      <option value="0">0 型</option>
    </select> <br/>
自我介紹: <br/>
<textarea name="AboutMe">
</textarea> <br/>

<input type="submit" value="送出"/><input type="reset" value="清除"/><br/>
</form>
```



The image shows a web browser window with a form titled "form.htm". The browser's address bar shows the file path "file:///D:/Dropbox/". The form contains the following fields and controls:

- 帳號: ccc
- 密碼:
- 信箱: ccc@gmail.com
- 生日: 2013/01/31
- 照片: 選擇檔案 1654.jpg
- 性別: 男 女
- 血型: A 型
- 自我介紹: (empty text area)
- Buttons: 送出 (Submit), 清除 (Clear)

表單的顯示結果

在上述的範例中，當 submit 類型的送出鈕被按下後，瀏覽器會將這些填寫的資訊，以第一章所說的 GET/POST 方式，發送給伺服器，如果 method 欄位是 GET，那麼會採用在 HTTP 表頭網址處傳送 `signup?user=xxx&password=xxx` 這樣的形式送出，這種方式會將密碼顯示在瀏覽器的網址列上，比較容易被看到，若 method 欄位是 POST，則會在 HTTP 表頭尾端加上 `user=xxx&password=xxx ...` 的資訊，不會在網址列上被看到。當然、如果有人監控網路上訊息的話，還是會看得到這些輸入資訊。

若要更安全，則必須採用 HTTPS 的 SSL 方式傳遞，這種方式會對訊息加密編碼，就比較不會有輸入訊息外洩的危險。

參考文獻

- W3School : HTML Tutorial -- <http://www.w3schools.com/html/default.asp>

CSS 版面設計

CSS 簡介

在前一章當中，我們介紹了 HTML 的語法，我們可以透過 HTML 的表格排出某些格式，但是卻很難讓整個頁面足夠美觀。

舉例而言，假如我們想讓表格的標頭欄以黑底白字的方式顯示，那麼就得靠 CSS 來幫忙了。

CSS 是 Cascading Style Sheets 的簡稱，中文可翻譯為「串接樣式表」，CSS 主要是用來設定 HTML 的顯示版面，包含「字體、大小、顏色、邊框、背景」等等，一個好的 CSS 可以讓您的網頁變得很美觀，而且不需要大幅修改網頁的內容，只要加入一個 CSS 引用即可。

當套用不同的 CSS 時，網頁的風格就可以產生完全不同的感覺，這讓你的網頁可以輕易的更換成不同的版型。事實上、當您在 blogspot 或 wordpress 網誌平台套用不同版型的時候，只不過是更換了 CSS 樣版而已。

在 HTML 當中，CSS 有兩種寫法，一種是內嵌式的寫法，另一種是外連式的寫法。

為了說明 CSS 的語法，先讓我們回顧一下，一個簡單沒有格式化的 HTML 表格，其寫法如下：

```
<html>
<head><meta charset='utf-8'></head>
<body>
<table>
<tr><th></th><th>欄 1</th><th>欄 2</th></tr>
<tr><th>列 1</th><td>1, 1</td><td>1, 2</td></tr>
<tr><th>列 2</th><td>2, 1</td><td>2, 2</td></tr>
</table>
</body>
</html>
```

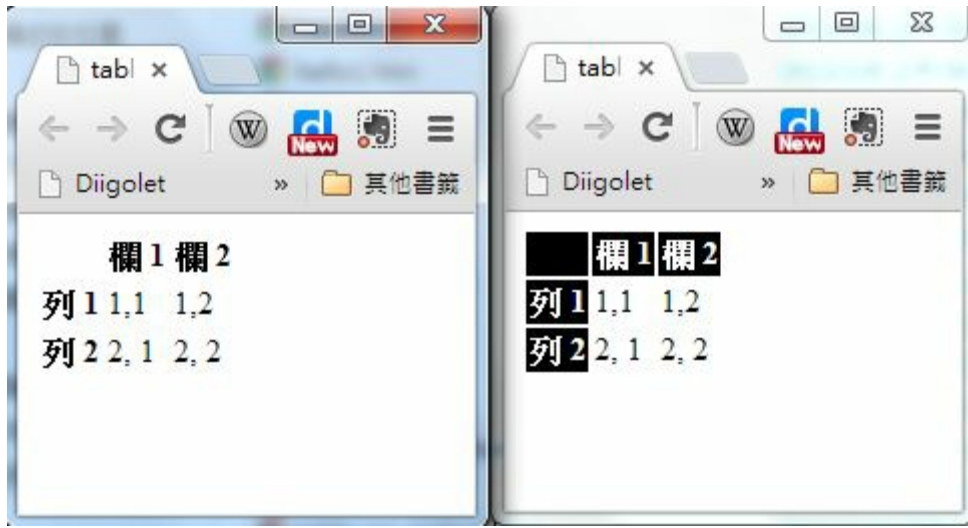
檢視檔案：<https://dl.dropbox.com/u/101584453/web/wp/code/table.htm>

如果我們將上述表格的 th 部分，都加上 style="background-color:black; color:white;" 這樣的 CSS 語法，那麼我們就會得到一個黑底白字的表格。


```
<html>
<head><meta charset='utf-8'></head>
<body>
<table>
<tr>
<th style="background-color:black; color:white;"></th>
<th style="background-color:black; color:white;">欄 1</th>
<th style="background-color:black; color:white;">欄 2</th>
</tr>
<tr>
<th style="background-color:black; color:white;">列 1</th>
<td>1, 1</td>
<td>1, 2</td>
</tr>
<tr>
<th style="background-color:black; color:white;">列 2</th>
<td>2, 1</td>
<td>2, 2</td>
</tr>
</table>
</body>
</html>
```

檢視檔案：https://dl.dropbox.com/u/101584453/web/wp/code/table_css_embed.htm

我們使用 CSS 排版前與排版後的結果可以對照如下：



表單的顯示結果

您可以看到在這種寫法當中，我們幾乎都一直在 th 重複撰寫 `style="background-color:black; color:white;"` 這一行文字，總共重複了 5 次，這顯然是很浪費時間的行為。

如果我們可以改用統一的方式，寫出「對於所有 th 而言，我都要用 `background-color:black; color:white;` 這樣的方式顯示」那不就簡單多了嗎？

是的、CSS 確實可以讓您這樣做，讓我們將上述範例修改成統一設定格式的版本，如下所示。

```
<html>
<head>
<meta charset='utf-8'>
<style>
th { background-color:black; color:white; }
</style>
</head>
<body>
<table>
<tr><th></th><th>欄 1</th><th>欄 2</th></tr>
<tr><th>列 1</th><td>1,1</td><td>1,2</td></tr>
<tr><th>列 2</th><td>2,1</td><td>2,2</td></tr>
</table>
```

```
</body>
</html>
```

上述統一將 CSS 寫在 `<head><style>...</style></head>` 裏的這種寫法，比起內嵌式的顯然簡短多了，這種寫法正是 CSS 的精華之所在。

但是、如果我們想要對整個網站的所有檔案，都套用同一種格式的話，那麼在每個 HTML 的頭部都要嵌入一整套 CSS 語法，那就不太方便了，萬一我們想要改變版面格式，不就每個 HTML 檔案都要更改，這顯然還不夠好用。

要解決這個問題，可以將表頭的 CSS 獨立到一個檔案中，然後再用連結的方式引用，這樣只要修改那個 CSS 檔案，整個網站的風格就跟著改變，所以上述檔案就可以改寫如下。

檔案：table_css.htm

```
<!DOCTYPE html>
<html>
<head>
<meta charset='utf-8'>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
<body>
<table>
<tr><th></th><th>欄 1</th><th>欄 2</th></tr>
<tr><th>列 1</th><td>1, 1</td><td>1, 2</td></tr>
<tr><th>列 2</th><td>2, 1</td><td>2, 2</td></tr>
</table>
</body>
</html>
```

檔案：mystyle.css

```
th { background-color:black; color:white; }
```

透過這種方式，我們就可以為網頁設計統一的風格，也比較容易更換網頁的排版風格了，這就是 CSS 語法的用途了。

當然、上述的網頁還不夠好看，如果我們將 mystyle.css 修改如下，您就會發現表格變得好看多了。

```
table { border-collapse: collapse; border: 1px solid #373737; }
th { background-color:black; color:white; padding:10px; margin:10px; }
td { padding:10px; margin:10px; }
```



The screenshot shows a web browser window with a single tab titled 'table_css.htm'. The address bar shows 'Diigolet' and a search icon. The main content area displays a table with a black header and white cells. The table has two columns and two rows of data. The header row has two columns labeled '欄 1' and '欄 2'. The first row of data has two cells labeled '1,1' and '1,2'. The second row of data has two cells labeled '2,1' and '2,2'. The table is styled with a solid black border and a white background for the cells.

	欄 1	欄 2
列 1	1,1	1,2
列 2	2,1	2,2

較優美的表單顯示結果

路徑, tag, id 與 class

即使有了可以統一套用的方式，有時我們還是會感覺到不方便，舉例而言，如果我們想要對某個表格套上 A 格式，然後在對另一個表格套上 B 格式，這時採用上述方法就不太夠用了。

為了解決這種問題，HTML 發展出了 id 與 class 這兩個屬性，我們可以根據 id 或 class 的名稱分別套用不同的 CSS 樣式。

舉例而言，以下網頁中有兩個 div 框，我們可以分別為這兩個框套用不同的格式，其中 classA 前面加了點符號，成為「.classA」，代表要比對的是 class 屬性，如果想要比

對的是 id 屬性，那麼就可以加上井字符號 # (像是範例中 #topbar 的情況)。

```
<html>
<head>
<meta charset='utf-8' >
<style>
#topbar { background-color:gray; color:blue; padding:10
px; margin:10px; }
.classA { background-color:black; color:white; padding:
10px; margin:10px; }
.classB { background-color:blue; color:yellow; padding:
10px; margin:10px; }
</style>
</head>
<body>

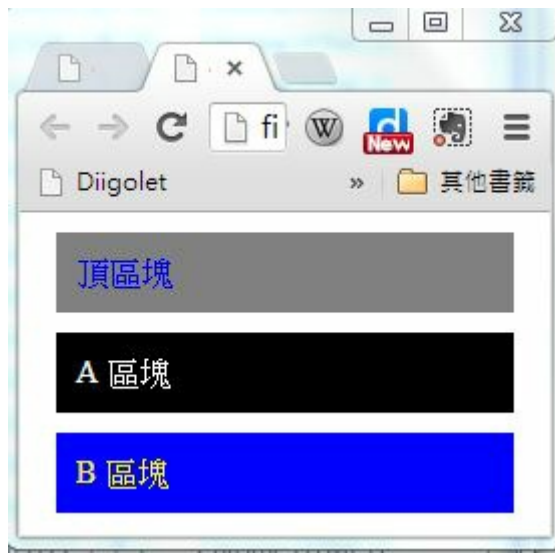
<div id="topbar">頂區塊</div>

<div class="classA">A 區塊</div>

<div class="classB">B 區塊</div>

</body>
</html>
```

檢視檔案：https://dl.dropbox.com/u/101584453/web/wp/code/div_css.htm



兩種 CSS class 的顯示結果

到目前，我們可以看到有三種 css 的指定方式，也就是「標記 tag、代號 id 與類別 class」，可以用 CSS 來定位。事實上、這些指定方式還可以互相串接，以下是一些範例：

```
.classA table { ... } // 套用到 class="class A" 內部的 table 標記
#topbar { ..... } // 套用到 id="topbar" 的元素上
#topbar table .classA a { ..... } // 套用到 table 內具有 class="classA" 類別裏的超連結 a 標記上。
```

更棒的是、這些選取方式之間還可以共用，只要利用逗點符號「,」分隔就可以了，以下是一些範例：

```
.classA, #topbar { ... } // 套用到 class="class A" 或 id="topbar" 的元素上
#topbar a, .classA a { ..... } // 套用到 id="topbar" 或 class=".classA" 裏的 a 元素上
```

一個實用的 CSS 的範例

如果您想進一步學習 CSS 的各個屬性與用法，請參考下列網址，我們將不一一說明

這些屬性的用法。

- <http://www.w3schools.com/css/default.asp>

在此、筆者將自己常用的一個 CSS 檔案列出，讀者觀察後可以自行複製修改使用。

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font: inherit;
    vertical-align: baseline;
    line-height: 160%;
}

h1, h2, h3, h4, h5, h6 {
    color: #333333;
    margin: 0;
    font-family: '標楷體', 'Times New Roman';
```

```
font-weight: bold;
}

p {
margin: 10px 0 15px 0;
font-size:100%;
}

li {
font-size:100%;
}

footer p {
color: #f2f2f2;
}

a {
text-decoration: none;
color: #007edf;
text-shadow: none;

transition: color 0.5s ease;
transition: text-shadow 0.5s ease;
-webkit-transition: color 0.5s ease;
-webkit-transition: text-shadow 0.5s ease;
-moz-transition: color 0.5s ease;
-moz-transition: text-shadow 0.5s ease;
-o-transition: color 0.5s ease;
-o-transition: text-shadow 0.5s ease;
-ms-transition: color 0.5s ease;
```

```
-ms-transition: text-shadow 0.5s ease;
}

table {
  border-collapse: collapse;
  border-spacing: 0;
  border: 1px solid #373737;
  margin-bottom: 20px;
  text-align: left;
  margin-left: auto;
  margin-right: auto;
}

th {
  padding: 10px;
  background-color: black;
  color: white;
}

td {
  padding: 10px;
  border: 1px solid #373737;
}

em { font-weight: bold; }

#topbar {
  margin: 0;
  padding: 1px;
  border: 0;
```

```
font: inherit;
vertical-align: baseline;
background-color:black;
color:white;
color:white;
width:95%;
text-align:right;
font-weight:bold;
}

#content {
  margin:10px;
  padding:10px;
}

pre {
  border: 1px solid #373737;
  background-color:#dddddd;
  color:#333333;
  font-size:medium;
  width:95%;
  padding:10px;
}

img {
  border: 1px solid #373737;
  margin-left: auto;
  margin-right: auto;
  display: block;
}
```



```
.figure .caption {
  text-align:center;
}

#footer {
  text-align:center;
  font-size:small;
  color:#666666;
  margin: 10px;
  padding: 10px;
}
```

參考文獻

- <http://www.w3schools.com/css/default.asp>
- <http://css.maxdesign.com.au/listutorial/>

JavaScript -- 讓網頁動起來

JavaScript 之所以重要，是因為 JavaScript 是瀏覽器當中規定的標準語言。這是因為早期的瀏覽器 Netscape 將 JavaScript 內建於其中，以變能夠進行一些互動程式的效果，因此才讓 JavaScript 成為瀏覽器的標準語言。

在瀏覽器中，要用 JavaScript 設計出互動效果，畢需借助 DOM (Document Object Model) 這個 HTML 的抽象化物件模型，這個模型將整個 HTML 文件視為一個樹狀結構的物件，於是 JavaScript 可以透過操控物件屬性的方式，達成某些互動性的顯示效果。這種互動性的顯示效果形成了瀏覽器特有的視覺化互動模式，可以說是近 15 年來 Web 的發展重點之所在。

接著、就讓我們透過一些簡單的程式範例，看看如何用 JavaScript 操控 DOM 物件模型，以便完成這種互動效果。

用 JavaScript 操控網頁

取得內容：innerText 與 innerHTML

以下範例會取得 hi 節點的 innerText 與 innerHTML 顯示出來，請觀察其不同點。

```
<html>
  <head>
    <title>節點存取示範</title>
  </head>
  <body>
    <div id = "hi"><b>你好!</b></div>
    <input type="button" value="hi.innerText"
      onclick="alert(document.getElementById('hi').
innerText)");
    <input type="button" value="hi.innerHTML"
      onclick="alert(document.getElementById('hi').
innerHTML)");
  </body>
</html>
```

顯示與隱藏 (Show and hide)

```
<html>
  <head>
    <title>範例 -- 顯示隱藏</title>
  </head>
  <body>
    <div id = "hi"><b>你好!</b></div>
    <input type="button" value="hi.show()" onclick="docume
nt.getElementById('hi').style.visibility='visible'");
    <input type="button" value="hi.hide()" onclick="docume
```

```
nt.getElementById('hi').style.visibility='hidden'");  
</body>  
</html>
```

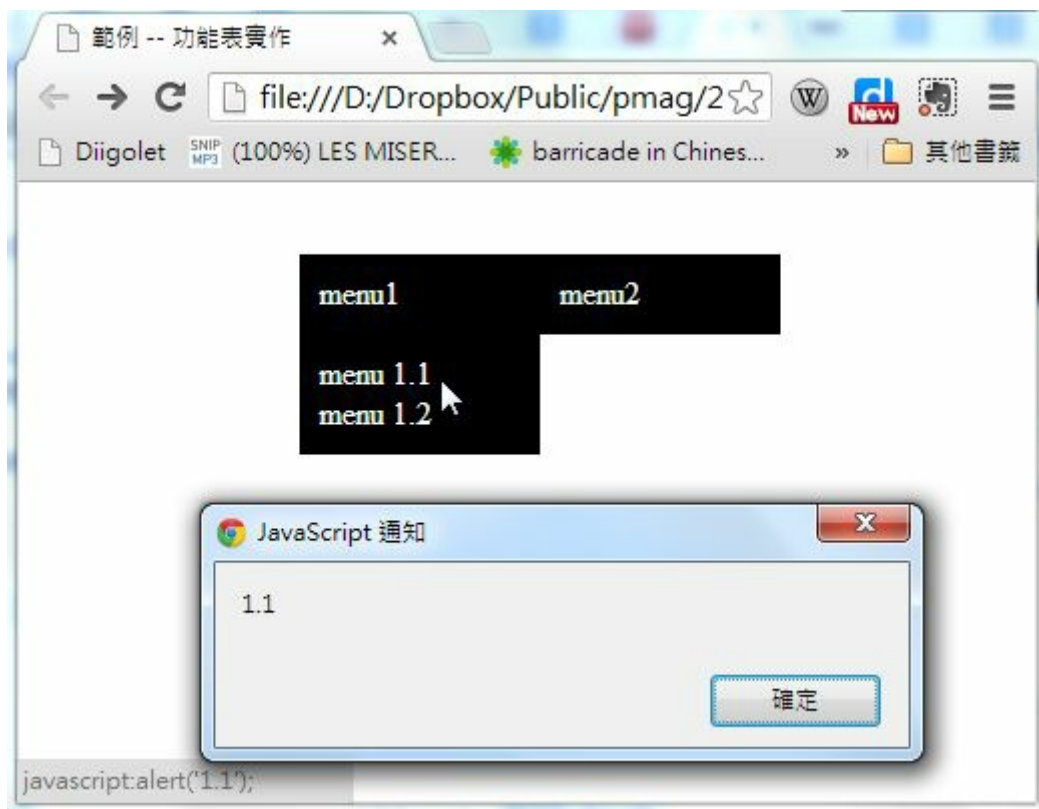
功能表程式

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 這樣的伺服器端 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。

在本節當中，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。

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-t
ype: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-t
ype: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 ... 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 ... 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。

小結

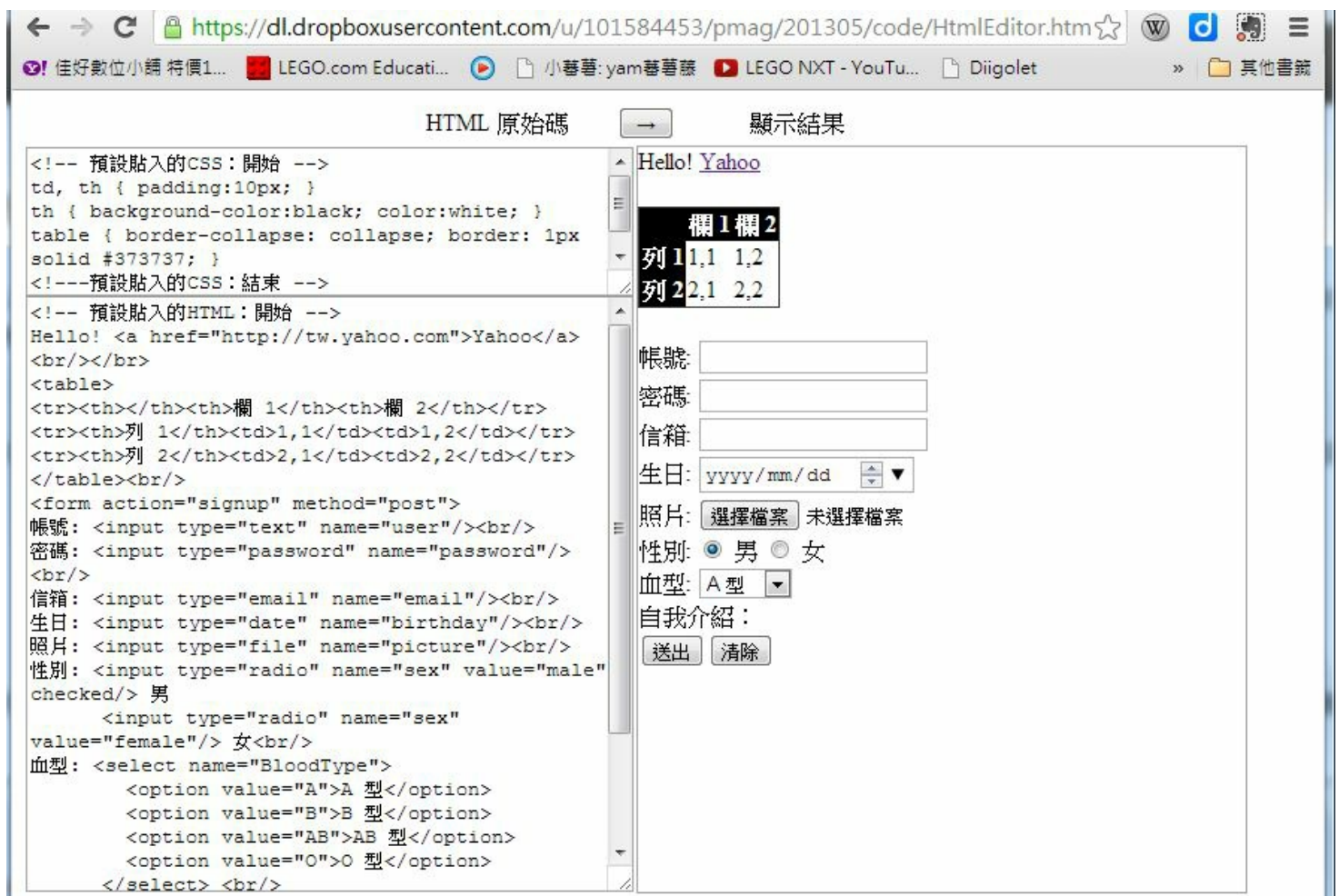
從上述範例中，您可以看到瀏覽器中的 JavaScript，通常是透過調整網頁某些項目的 CSS 屬性，以達成互動性的功能，這種互動網頁技術，事實上是結合了 HTML+CSS+JavaScript 等技術才能達成的功能，因此這三項技術在瀏覽器當中幾乎是合為一體、可以說是缺一不可的。

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！

HTML 編輯器

在本節當中，我們將透過 JavaScript 設計一個 HTML 編輯器，讓您可以直接在瀏覽器當中看到 HTML 網頁的呈現結果，筆者認為這個範例對學習動態網頁設計是一個非常簡單、卻又很有啟發性的程式。因為他很明顯的展現了動態網頁的原理。

以下是該 HTML 編輯器的執行結果，當我們在 CSS 與 HTML 區塊輸入對應的原始碼之後，就可以按下「→」按鈕，然後在呈現的 `<div id="showbox" ...</div>` 區塊看到兩者搭配時的呈現結果。



HTML 編輯器的執行畫面

您也可以點選下列連結以實際檢視該網頁：

- HTML 編輯

器: <https://dl.dropboxusercontent.com/u/101584453/pmag/201305/code/HtmlEditor.htm>

以下是這個網頁的原始 HTML 程式碼，其中有兩大段是我們預設填入的 CSS 與 HTML 原始碼，這兩段原則上可以去掉，但是為了測試方便起見，我們就留在檔案中，請讀者閱讀的時候仔細區分之。

```
<html>
<style>
textarea, #showbox { border: 1px solid #9f9f9f; }
</style>
<style id="showboxstyle">
</style>
```

```
<script type="text/javascript">
function convert() {
    var cssbox = document.getElementById("cssbox");
    var editbox = document.getElementById("editbox");
    var showbox = document.getElementById("showbox");
    var showboxstyle = document.getElementById("showboxstyle"
);
    showbox.innerHTML = editbox.value;
    showboxstyle.innerHTML = cssbox.value;
}
</script>
<body>
<form>
<table width="95%" style="border-collapse: collapse; border: 0px;"><tr>
<tr><td colspan="2" style="text-align:center">HTML 原始碼
    <input type="button" value=" → " onclick="JavaScript:
convert()"/>    顯示結果</td></tr>
<td width="50%">
<textarea id="cssbox" style="width:100%; height:100px;" >
<!-- 預設貼入的CSS: 開始 -->
td, th { padding:10px; }
th { background-color:black; color:white; }
table { border-collapse: collapse; border: 1px solid #3737
37; }
<!--預設貼入的CSS: 結束 -->
</textarea>
<textarea id="editbox" style="width:100%; height:400px;">
<!-- 預設貼入的HTML: 開始 -->
Hello! <a href="http://tw.yahoo.com">Yahoo</a><br/></br>
```

```
<table>
<tr><th></th><th>欄 1</th><th>欄 2</th></tr>
<tr><th>列 1</th><td>1, 1</td><td>1, 2</td></tr>
<tr><th>列 2</th><td>2, 1</td><td>2, 2</td></tr>
</table><br/>
<form action="signup" method="post">
帳號: <input type="text" name="user"/><br/>
密碼: <input type="password" name="password"/><br/>
信箱: <input type="email" name="email"/><br/>
生日: <input type="date" name="birthday"/><br/>
照片: <input type="file" name="picture"/><br/>
性別: <input type="radio" name="sex" value="male" checked/
> 男
      <input type="radio" name="sex" value="female"/> 女<b
r/>
血型: <select name="BloodType">
      <option value="A">A 型</option>
      <option value="B">B 型</option>
      <option value="AB">AB 型</option>
      <option value="O">O 型</option>
    </select> <br/>
自我介紹: <br/>
<input type="submit" value="送出"/><input type="reset" val
ue="清除"/><br/>
</form>
<!-- 預設貼入的HTML: 結束 -->
</textarea>
</td>
<td>
<div id="showbox" style="width:100%; height:500px;">
```



```
</div>
</td>
</tr></table>
</form>
</body>
</html>
```

程式解析

上述程式當中有三個主要的區塊，分別是：

1. CSS 填入區塊： `<textarea id="cssbox" ...</textarea>`
2. HTML 填入區塊： `<textarea id="editbox" ...</textarea>`
3. HTML 顯示區塊： `<div id="showbox" ... </div>`

此網頁的核心程式部分真的很簡單，只有如下短短的一小段：

1. 利用 `var editbox = document.getElementById("editbox");` 取得 HTML 區塊內容
2. 利用 `showbox.innerHTML = editbox.value;` 這個指令將該 HTML 原始碼填入 `showbox` 當中

這樣就完成顯示 HTML 的動作了。

```
function convert() {
    var cssbox = document.getElementById("cssbox");
    var editbox = document.getElementById("editbox");
    var showbox = document.getElementById("showbox");
    var showboxstyle = document.getElementById("showboxstyle");
};
showbox.innerHTML = editbox.value;
showboxstyle.innerHTML = cssbox.value;
}
```

但是這樣作並沒有加入 `cssbox` 的內容到 HTML 當中，因此我們加入了下列原始碼：

1. 在整個網頁的頭部事先用 `<style id="showboxstyle">...</style>` 這個標記加入一個 CSS style 顯示區塊
2. 利用 `showboxstyle.innerHTML = cssbox.value` 這個指令將 `cssbox` 的內容填入到該表頭的 style 區塊中

這樣就達成了套用 CSS 內容到網頁中的目的，完成了整個 HTML 編輯器的功能。

小結

在本節中，我們用非常簡單的程式，建構了一個 HTML 編輯器。事實上我們只不過是把網頁內容從編輯區域移動到顯示區域，然後瀏覽器就會自動解釋這些內容進行呈現了。

參考文獻

- <http://stackoverflow.com/questions/1720320/how-to-dynamically-create-css-class-in-javascript-and-apply>
- <http://dev.opera.com/articles/view/dynamic-style-css-javascript/>

科技人小故事

與大師相遇：Tim Burner Lee



圖、Tim Berners-Lee 2012 年的照片

來源：http://en.wikipedia.org/wiki/Tim_Berners-Lee#mediaviewer/File:Tim_Berners-Lee_2012.jpg

Tim Burner Lee 被尊稱為 Web 之父可以說是實至名歸，因為他發展出了 HTTP+HTML+URL 等技術，並創建了全世界第一個伺服器與陽春版的瀏覽器，雖然後來 MOSAIC 在瀏覽器上做得更好，但整個技術的架構確實是由 Tim Burner Lee 所奠定的基礎。

1980年6月-9月間，柏納-李在CERN（歐洲核子研究組織）擔任獨立承辦人時，他提出了建立一個以超文字系統為基礎的專案，以方便研究人員分享及更新訊息，同時他建立了一個原型系統，叫 ENQUIRE。

1980年，Tim Burner Lee 離開 CERN 轉而任職於約翰·普爾圖形電腦系統有限公司，該公司位於英格蘭伯恩茅斯。在這間公司里，他參與的計劃是一個遠端程序呼叫，從而獲得了電腦網路經驗，1984年，Tim Burner Lee 以正式員工的身份重返 CERN。

在1989年的時候，CERN 是全歐最大的網際網路節點，並因此看到了將超文字系統與網際網路結合在一起的機會。他想到：「我只要把超文字系統和傳輸控制協定、域名系統結合在一起，就能得出全球資訊網了！」。

1989年3月，他寫下了他的初步構想。次年，在羅伯特·卡里奧的幫助下，作出了一

個修訂版，並得到主管麥克·森德爾（Mike Sendall）認可。Tim Burner Lee 以製作 ENQUIRE 系統時的基本概念建立全球資訊網，並設計製作出世上第一個網頁瀏覽器。同時，他也設計了世上第一個網頁伺服器，以下是羅伯特·卡里奧回憶當時的一段訪談：

"Mike Sendall buys a NeXT cube for evaluation, and gives it to Tim Berners-Lee. Tim's prototype implementation on NeXTStep is made in the space of a few months, thanks to the qualities of the NeXTStep software development system. This prototype offers WYSIWYG browsing/authoring! Current Web browsers used in "surfing the Internet" are mere passive windows, depriving the user of the possibility to contribute. During some sessions in the CERN cafeteria, Tim and I try to find a catching name for the system. I was determined that the name should not yet again be taken from Greek mythology. Tim proposes "World-Wide Web". I like this very much, except that it is difficult to pronounce in French..." by Robert Cailliau, 2 November 1995.

1991年8月6日，世上第一個網站在 CERN 建成並上線：

Info.cern.ch was the address of the world's first-ever web site and web server, running on a NeXT computer at CERN. The first web page address was <http://info.cern.ch/hypertext/WWW/TheProject.html>, which centred on information regarding the WWW project. Visitors could learn more about hypertext, technical details for creating their own webpage, and even an explanation on how to search the Web for information. There are no screenshots of this original page and, in any case, changes were made daily to the information available on the page as the WWW project developed. You may find a later copy (1992) on the World Wide Web Consortium website.

這解釋了 World Wide Web 是什麼，用戶如何使用瀏覽器，如何建立網頁伺服器。

接下來、全世界都被捲入到 World Wide Web 大開發的浪潮中，而 Tim Burner Lee 也因此而成了 Web 之父。

後來、在 1994年，Tim Burner Lee 在麻省理工學院創辦了 World Wide Web Consortium，成為全世界制定 Web 標準規格的主要機構，很多與 Web 有關的組織與公司都會透過 World Wide Web Consortium 參與 Web 新規格的制定過程。

參考文獻

- [維基百科：提姆·柏納-李](#)
- [Wikipedia:Tim Berners-Lee](#)

影音頻道

電腦網路的概念與歷史 -- Internet 與 Web

在本期的主題當中，我們已經用了很多文字來說明 Internet 與 Web 這兩個相關技術的概念與歷史了，但是有些事情用影片的方式傳達會更容易，因此筆者上網找了一些有關 Internet 與 Web 的影片。

以下是筆者覺得還不錯的影片，提供給您參考！

- How the Internet Works in 5 Minutes -- http://youtu.be/7_LPdttKXPc
- History of the Internet -- <http://youtu.be/9hIQjrMHTv4>
- What is the World Wide Web? - Twila Camp -- <http://youtu.be/J8hzJxb0rpc>
- How the World Wide Web just happened - Tim Berners-Lee -- <http://youtu.be/yF5-6AcohQw>
- World Wide Web Turns 25: Interview with inventor Sir Tim Berners-Lee -- <http://youtu.be/loi6PYaRqHA>

科學原理

電腦的數學基礎：二進位系統與運算

十進位

小學所教的加減乘除方法都是以十進位為基礎的，舉例而言、十進位的 358 其實代表的是 $3*100+5*10+8*1$ 這樣以 10 為基礎的表達系統，如果寫成直式或許讀者會看得更清楚。

300		$3*100$
50		$+ 5* 10$
8		$+ 8* 1$
+ -----	=	-----
358		$3*100+5*10+8*1$

二進位

但是在電腦當中，十進位並不好用，二進位才是比較適合電腦的表達方式，以下是二進位裏前幾個位元所代表的意義。

0 0 0 1	代表 1
0 0 1 0	代表 2
0 1 0 0	代表 $4 = 2*2$
1 0 0 0	代表 $8 = 2*2*2$

於是，二進位的 1101 所代表的，其實是如下的一個數字：

1000		8
100		4
1		1
+ -----	=	+ -----

二進位加法

當我們用十進位做加法運算的時候，每當數字到達十就必須進位，這也正是十進位這個名詞的意義，以下是一個十進位加法的範例。

進位	11
A	358
B	74
+-----	-----
A+B	432

同樣的、當我們用二進位作加法的時候，只要數字到達 2 就必須要進位，以下是一個二進位加法的範例：

進位	1 1 1 1 1
A	0 1 1 0 1
B	1 0 1 1 1
+-----	-----
A+B	1 0 0 1 0 0 = 十進位的 36

小結

當您可以理解二進位與其加法之後，就可以進一步理解「用二補數表示負數、二進位減法、乘法與除法」等等，未來我們將會進一步討論這些主題。

現在我們已經學會二進位的表示法，還有二進位的加法了，於是我們可以開始設計二進位加法的電路了，讓我們回到邏輯閘的世界吧!

參考文獻

- [維基百科：邏輯閘](#)
- [維基百科：加法器](#)
- [維基百科：二進制](#)

- [Wikipedia:Binary Number](#)

科技人專欄

組成電腦的基礎元件 -- 邏輯閘

在 [上期的少年科技人雜誌] 當中，我們探討了「電腦的歷史、工業與結構」這個議題，並且在 [透視電腦的內部結構](#) 這篇文章中看到了電腦的基本組成元件，那就是邏輯閘。

在接下來的幾期當中，我們將逐步的從邏輯閘開始，解說「電腦是如何用邏輯閘所組成的」這個問題。

組成電腦的基本邏輯閘包含 AND、OR、NOT、XOR 等四種，其中 AND、OR、NOT 三種就可以組成所有電路，包含 XOR，但是加上 XOR 時會比較方便，因為在有些電路上用 XOR 可以大量減少邏輯閘的數量。

基本邏輯閘 (Logic Gate)

反閘 (NOT) 的圖示與功能如下，其輸入輸出都只有一條，當輸入 0 的時候就輸出 1，而輸入 1 的時候就輸出 0。

類型	ANSI及IEEE標準	IEC標準	名稱	短釋	邏輯函數表示	真值表								
NOT			「非」門/反相器/「反」閘/逆變器	輸入的高低狀態會逆轉。	\bar{A}	<table border="1"><thead><tr><th>輸入</th><th>輸出</th></tr></thead><tbody><tr><td>A</td><td>NOT A</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></tbody></table>	輸入	輸出	A	NOT A	0	1	1	0
輸入	輸出													
A	NOT A													
0	1													
1	0													

圖、反閘 (NOT)

及閘 (AND) 的圖示與功能如下，其輸入有兩條，輸出只有一條，當輸入 11 的時候才會就輸出 1，其他三種情況則一律輸出 0。



圖、及閘 (AND)

換言之、AND 是在 A B 兩者都為 1 的時候才會輸出 1，這和邏輯學上的「且」(AND) 相同，當我們說「他高且帥」的時候，代表他同時具有「高」和「帥」兩個特性。

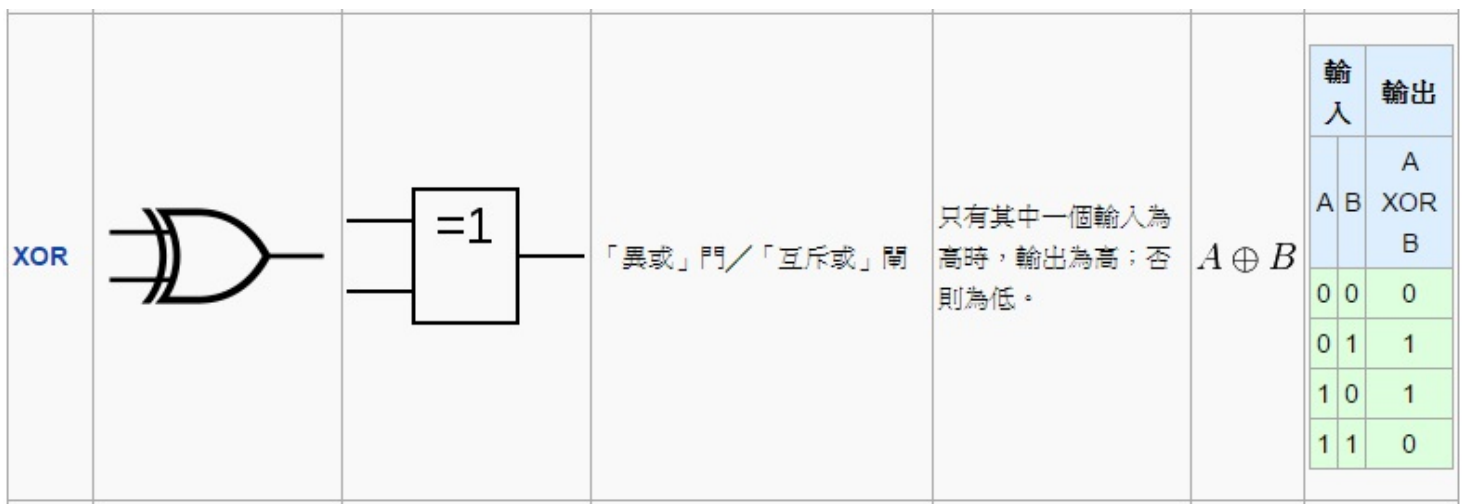
或閘 (OR) 的圖示與功能如下，其輸入有兩條，輸出只有一條，當輸入 00 的時候才會就輸出 0，其他三種情況則一律輸出 1。



圖、或閘 (OR)

換句話說、只要 A B 兩者當中有一個為 1，那麼輸出就會是 1，這和邏輯學上的「或」(OR) 相同，當我們說「高或帥」的時候，代表他只要具備「高」或「帥」其中一個特性就滿足了條件。

互斥或閘 (XOR) 的圖示與功能如下，其輸入有兩條，輸出只有一條，當兩條不一樣的時候就輸出 1，兩者一樣的時候則會輸出 0。



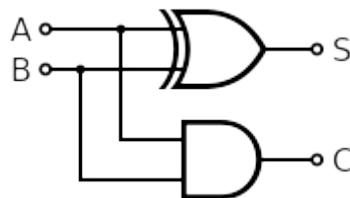
圖、互斥或閘 (XOR)

有了上述邏輯閘之後，我們就可以用這樣的閘來組成電腦當中最核心的一個運算，那就是加法了。

但是、要理解如何用邏輯閘組成加法運算之前，我們必須先了解二進位這個概念。

半加器與全加器

將兩個輸入相加的電路，稱為「半加器」(half adder)，而將三個輸入相加的電路，則稱為「全加器」(full adder)。



圖、半加器

半加器的電路如上圖所示，而其輸入輸出的真值表如下所示：

$A + B \rightarrow S, \text{ 進位 } C$
$0 + 0 \rightarrow 0 \quad 0$
$0 + 1 \rightarrow 1 \quad 0$
$1 + 0 \rightarrow 1 \quad 0$
$1 + 1 \rightarrow 0, \quad 1 \text{ (因為 } 1 + 1 = 2 = 0 + (1 \times 2) \text{)}$

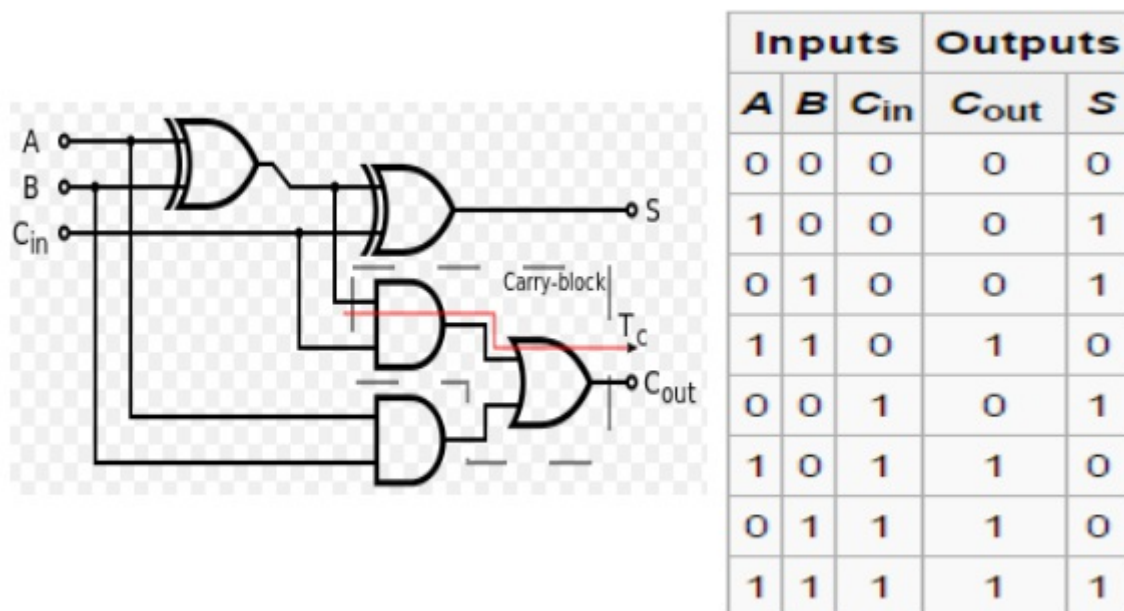
上述的輸出 S 只有在 A, B 兩者不同的時候才會是 1，這完全符合 XOR 電路的行為模

式，因此我們可以用一個 XOR 電路輕易地完成 S 的輸出動作。

而 C 則是在 A, B 兩者均為 1 的情況下才會是 1，否則就會是 0。這與 AND 閘的真值表完全一致，因此我們只要用一個 AND 閘就可以做到 C 的功能了。

但是、雖然我們想做的事情是兩組二進位數字的加法，但是由於相加之後可能會產生進位，因此必須要考慮到三個位元相加的情況，這時就需要將半加器進一步擴充為全加器，以便進行三位元相加的運算。

以下是全加器的電路圖與真值表，其中的輸出 S 只要用兩個 XOR 閘就能做到，因為 S 只有在奇數個 1 的時候才會輸出 1，偶數個 1 的時候就會輸出 0，這種組合完全符合 n 輸入的 XOR 閘之行為。



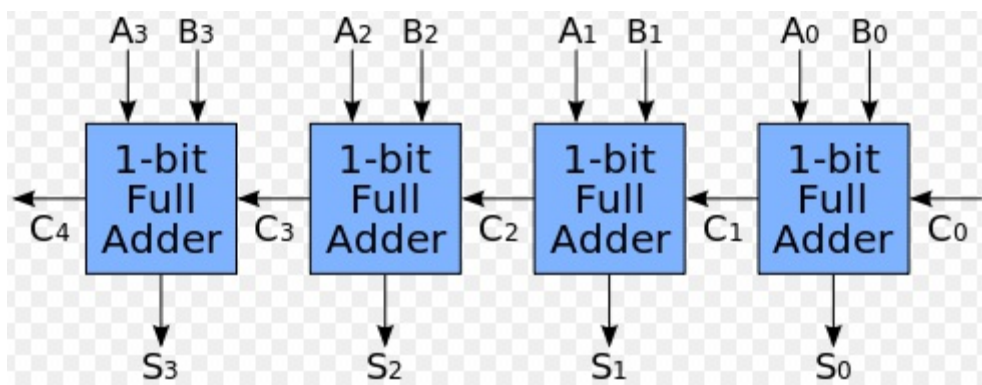
圖、全加器的電路與真值表

但是對於輸出 C 而言，則需要兩個 AND 閘和一個 OR 閘連接之後才能達成，您可以根據上面的 AND, OR 之真值表定義，驗證一下該電路的輸出是否真的符合上圖真值表的要求，這會是一個很好的練習。

在此、我們直接提出了全加器的電路圖，而沒有展示這個電路的「設計過程」，因為這種設計過程牽涉到一種稱為「卡諾圖」的電路化簡工具，

- [維基百科: 卡諾圖](#)
- [Wikipedia:Karnaugh_map](#)

有了上述的全加器電路之後，我們就可以任意地進行三個位元的加法，於是只要我們將這些全加器一個接一個的串起來，並將進位輸出到下一個全加器當中，就可以任意地做出 n 位元的加法器電路了，以下是一個四位元加法器的圖示。



圖、四位元加法器

當您能做出 n 位元加法器時，應該就可以理解為何電腦的基本組成會是「邏輯閘」了，因為我們可以輕易地串接出 32 位元的加法器，當然也可以用類似的原理做出「減法器、乘法器、除法器」，然後做出整顆「中央處理器」(CPU)，甚至是整台電腦的數位電路了。

參考文獻

- [維基百科：邏輯閘](#)
- [維基百科：加法器](#)

雜誌訊息

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，請盡可能用 markdown 編輯好您的稿件，並於每個月 25 日前投稿到 [少年科技人社團](#) 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。

智財權注意事項

當您投稿時，最重要需注意的一件事情，是智慧財產權的問題。

您必須確定投稿的文章沒有侵權的問題！

如果文章與圖片完全是您自行製作的，那應該不會侵權。但是如果您有使用網路上的圖片或修改網路上的文章，請務必採用沒有侵權疑慮的來源，像是 [維基百科](#) 或者 [創作共用](#) 授權的文章，並且標明修改來源。

舉例而言，如果您修改自維基百科，請標上「本文修改自維基百科」，如果是其他「創作共用」文章或圖片，則需要附上原始文件或圖片的連結。

以下是「創作共用」文章的授權類型，您可以在標識姓名來源後採用「姓名標示（BY）」與「姓名標示（BY）-相同方式分享（SA）」這兩類授權的文章，也可以在特別標示授權後採用「姓名標示（BY）-非商業性（NC）」或「姓名標示（BY）-非商業性（NC）-相同方式分享（SA）」的文章，但是不要採用有「禁止改作（ND）」的文章來進行衍生創作。

授權條款	BY	NC	ND	SA
姓名標示（BY）				
姓名標示（BY）-相同方式分享（SA）				
姓名標示（BY）-禁止改作（ND）				
姓名標示（BY）-非商業性（NC）				
姓名標示（BY）-非商業性（NC）-相同方式分享（SA）				
姓名標示（BY）-非商業性（NC）-禁止改作（ND）				

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 [少年科技人社團](#) 或 [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	http://www.mncf.org/ lynn@mncf.org	顫顏患者 (如唇顎裂、小耳症或其他	銀行：009彰化銀行民生分行

夫顛顏基金會	02-27190408分機 232	罕見顛顏缺陷)	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