東南技術學院

機械工程系

探就 WWW

指導導師: 陳明洲

製作學生： 林建柱　徐金楷　范智豪

劉智義　連志偉　黃國峰

許鴻琳　曾國書　林宏隆

簡介:什麼市網際網路呢？什麼是 HTML？什麼是資料庫？什麼是 PHP？在我們

的報告中可以清楚的了解並且知道這一些日常生活中經常用到應用軟體，

網際網路只經過連線而把兩甚至兩部上的電腦連接在一起互相換交資訊。

。 。

HTML 是指專為設計網頁所開發出的程式語言，其基本架構必須要了解，

因為要把一些特殊效果放在你的網頁上時，就必須要了解 HTML 的基本

架構才有辦法完成.....。

HTML 簡介

HaperText Makeup Language(HTML)，中文稱為<超文件標記語言>，是一種專為設計網頁所開發出的程 式語言，雖然使用 FrontPage 設計網頁，都會自動將編輯完成的網頁轉成程式，並不需要知道 HTML 就可以 製作網頁，不過 HTML 的基本架構還是必須要瞭解，像是把一些特殊效果放在你的網頁上時，就必須要瞭解 HTML 的基本架構才有辦法完成，其實只是架構而已，非常容易的。

HTML 基本架構如下：

```
<HTML>
<HEAD>
.
.
</HEAD>
<BODY>
.
.
.
.
</BODY>
</HTML>
```

所以 HTML 基本上是由多種不同的標籤所組成（包含在 <HTML> ..... </HTML> 標籤裡），在 <HEAD> ..... </HEAD> 標籤裡主要是放一些網頁的相關資訊，如標題等等，而網頁的內容全部都放在 <BODY> ..... </BODY> 裡，而你只需要 Notepad 或純文字檔編輯器就可以，記得要把副檔名存為 .htm 或是 .html。以下是一個完整 HTML 文件範例：

```
<HTML>
<HEAD>
<TITLE>康老師工作室</TITLE>
</HEAD>
<BODY BACKGROUND="bg.gif" TEXT="black" >
<P>Hello! This is an example of HTML page!!</P>
</BODY>
</HTML>
```

在 <HEAD> ......</HEAD> 標籤裡，有個標籤(<TITLE> ..... </TITLE>)就是網頁的標題，會顯示於網頁的左上角，而標籤(<BODY> ..... </BODY>)就是網頁的 內容，所以 Hello! This is an example of HTML page!!這行字會顯示於網頁中。

DHTML 簡介

Dynamic HTML 簡稱為 DHTML（俗稱動態網頁。），是最近興起的網路技術，它不像 Flash 和 Javascript 一樣只是一種特有的技術，它可以說是結合了許多技術所賜予的稱號罷了，像是 style sheets、DOM、scripting language 等等。像水之精靈（那個飛來飛去的東東就是啦！）就是用了這技術所做成的。

DHTML 除了可以讓圖片在螢幕上飛來飛去之外，也可與使用者互相對應，像是你把滑鼠移到圖片上，圖片會換成另一張，還可發出聲音及超連結等。DHTML 的好處就是你不須要外掛程式，就可以做出互動式的網頁，更富於吸引力，目前 DHTML 只支援 IE4 和 Netscape4 以上的瀏覽器，而且這兩種瀏覽器所翻譯出來的效果又不盡相同或者只適用於一種，想要看看用 DHTML 所做的網頁，
http://www.bratta.com/，此站把 DHTML 的功能都發揮出來！

§ Dynamic HTML

Dhtml (Dynamic Html) 主要乃用來補足一般 HTML 無法達到的疊層與動態效果。DHTML
一般稱為動態網頁技術。而這裡的動態不並是指網頁畫面會移動稱為動態網頁，而是
指 HTML 配合 Javascript，其中的語法參數可以改變。

於 IE 的主要用法：乃用 Embeded 插入式 Style Sheet 在 head 中定義，但定義名稱前面加個 "#"

```
<head>
<style type=text/css>
```

```
<!--
#活頁名稱 {position:absolute; visibility:visible; z-index:5; top:200px; left:80px}
//-->
</style>
</head>

<body>
<div id=活頁名稱> position-block(活頁)前後用 div 夾起來，注意是用 id，而非 class</div>
```

-- position : absolute/relative (絕對位置或相對位置)
-- visibility : visible/hidden (顯示圖層或不顯示圖層)
-- z-index 主要作用在於 Cascading 疊層，數字越大位於越外層
  疊層效用使得網頁變成動態"Dynamic"效應，即可彈性調整參數
-- top : xx px(圖層距 window 上面距離，可為 px 值或螢幕高的百分比%)
-- left : xx px(圖層距 window 左邊距離，可為 px 值或螢幕高的百分比%)

※注意：position-block 稱活頁，乃圖層物件。class 用在排版；id 用在活頁。
※注意:直至目前為止許多 Dhtml 在 IE 與 Netscape 仍無法看到完全一樣的效果。

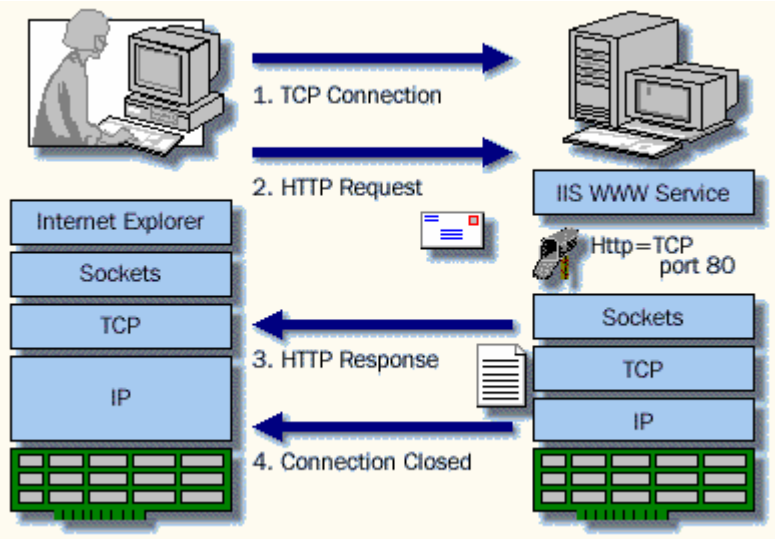
HTTP 通訊協定

Web 應用程式利用 Hypertext Transfer Protocol(HTTP)傳輸協定來聯繫瀏覽器與伺服器之間的溝通。

當使用者要求一個網頁時, 瀏覽器產生一個 HTTP 的需求訊息並將它送往伺服器端。伺服器會產生一個 HTTP 的回應訊息傳回給瀏覽器。在這個回應訊息中包含 HTML 文件。

HTTP 會談(Session)

下圖簡單說明一個 HTTP 會談的過程



各步驟描述如下

1. 瀏覽器與伺服器之間產生一個 TCP/IP 的連結(Connection)
2. 瀏覽器包裝一個對伺服器上 HTML 文件的需求, 並透過 TCP/IP 連線將訊息傳遞到伺服器。訊息的第一行會包含 HTTP 的需求方法。如果是一個簡單的網頁需求，會用 GET 方法。
3. 伺服器接收到一個 HTTP 需求，並以需求行內含要求的方法處理。
4. 伺服器接著傳回 HTTP 回應訊息。部分的回應訊息是包含顯示資料是否成功地照使用者所提的需求完成的狀態列資料。
5. 當瀏覽器收到 HTTP 回應訊息，TCP/IP 連結會被中斷， HTTP 會談會結束。

如果要求的 HTML 文件包含內定的物件，如圖檔，瀏覽器收到 HTML 文件之後會為每一個物件建立需求。例如，若網頁包含三個 GIF 圖檔，一個背景音樂以及一個 ActiveX 控制項，則會有六個 HTTP 會談會被建立以存取整個網頁。五個需求是用來要求內含物件，一個是為了網頁本身。

2.與資料庫的整合

<資料庫的起源與定義>

自從電腦發明以來,人們首先裡用他來處理科學科學上繁雜的計算問題.例如美國的太空計畫中,利用電腦快速的計算能力,可以將太空船升空,續航,以及回返地球的美一個細節都快速而且正確的掌握住.若沒有電腦的幫助,就沒有 1960 年代的太空探險成果.也因此,電腦或是電腦工業就在這種環境之中漸漸成熟.

到了電腦已經廣泛的使用在科學的用途上之後,科學家們就開始慢慢的將他推向其他應用領域之上.其中讓大家最快就能夠聯想到的方向之一,就是利用電腦來處理大量的資料.依據統計,人類在每六至七年當中就會產生出一倍的資料量.依照這個估計,如果沒有一種有效的方法及快速的的工具來幫助企業或個人,資料的運用及傳播將受到非常大的限制.因此,大量而又缺乏秩序的資料被電腦以及相關的軟體分析,歸類還有整理之後,便為可被利用的資訊.在此,我們對資料裡有了第一步的認識,也就是說,目前世界上充滿各種不同的資料種類及資料內容,但是若想能將這些資料變成有用的資訊,則必須依靠一套高效率的工具及方法.在工具方面,目前各種類型的電腦,從大型主機,工作站,以至於個人電腦都可以針對不同的需求,提供一套資料處理的有效工具.

在下面的資料中,將仔細介紹及分析各種不同的處理方法,首先將討論資料的主要重點?或者說,一套資料處理系統可以提供使用者什麼樣的幫助?

1.快速處理資料的能力

對於各種不同的資訊用而言,快速的資料分析及處理能力,將提供更精確,更快速的支援服務六如當電話使用者撥出一通長途電話時,在短短一,二秒中,資訊系統將要辨識撥話端的電話號碼,紀錄開始的通話時間,檢查發話端的設定特性或特殊服務,收集受話端的電話號碼並迅速的決定與哪一台長途電話交換機連線,等待長途交換機的回應後在回覆發話端.像這樣複雜的處理程序是無法利用人力所取代的工作.因此,一套快速處理資料的系統,才是提供快速及廉價電信服務的基礎.

2.可靠的處理資料的能力

利用電腦來取代人的工作時,他可以避免電許多不該有的錯誤,也就是說電腦在處理大量資料時,不會有疏忽,疲倦,分心等人為的弱點.雖然目前電腦仍然無法取代人腦的思想,創造等能力,但就其能忠實及可靠的處理大量資料的特性,仍然是我們最為依賴資料處理系統的地方.

3.可裝載大量資料的能力

依照目前人類所產生ㄉ資料速度來看,不僅資料本身在快速成長,

並且資料所演深出來的資料也藉著電腦資訊的成長而大量產生.如果完全依賴傳統的人工資料管理方式來處理它們,我們首先將面臨到資料保存的嚴重問題,例如世界各地的大型圖書館中,如何妥善保存不斷增加的文件及書籍之料就是其所面對的最大挑戰之一.就以目前的電腦系統來看,硬碟及光碟的儲存能力不斷的在提升,而其價格卻不斷的下降中,也就是說這些資料儲存設備,已經具備了作為永久保存資料的工具之一.此外,由於光碟將漸漸取代紙及印刷為主的資料出版方式.這不僅有助於資料的靈通,更可以減低自然生態的破壞.

4.價格低廉的資料處理方式

以最近數十年的變化來看,以人工來處理資料的成本,隨著通貨及薪資的不斷上升,使得企業花費在這的比例也不斷的上升中,相對來看,企業資料處理成本卻漸漸下降之中.以一位中等薪資的資料處理人員來看,企業大約每年要花費百萬元台幣左右人事成本,但目前一套值幾百萬的資料處理系統所享受到的生產力,是束倍於同等值人力的花費成本,這也是最近美國企業在不斷的裁減工作人力之下,卻扔然享有高度生產力的原因.有專家預測如果企業無法多多利用廉價且有效率的資訊系統來代替傳統人工的話,企業將在競爭之中被淘汰出局.

5.高度的資料互想及交流

如果資料的價值是可貴的話,則提高資料的共享及暢通資料交流的管

道就更可以發揮資料的價值.例如個人利用資料的共享來避免知識的差距,而企業及政府藉著資料的共享及交流來避免應協調不足所導至的損失.最明顯的例子就是'軼路技術的發展對個人而言,它提升了收集資料的能力及減低資料收集所花的時間及金錢;對企業而言,利用網路來達到高度的資料共享,使得企業之中的每一份子,都可以接觸到正確和相同的資料來源,以發揮團隊的力量.類似這種軟體及系統被稱群組軟體,例如 IBM的 Lotus Notes 以及近來流行的 Internet 等就都是注重其流程整合資料共享及高度流通的特性.

上面所談論的重點間接也表達出一樣事實,就是人們所需要的資訊和他每天所能得到的資料,仍然有一段距離,而這距離是需要功能強大的處理器,如電腦,來滿足人們對資訊的真正需求.

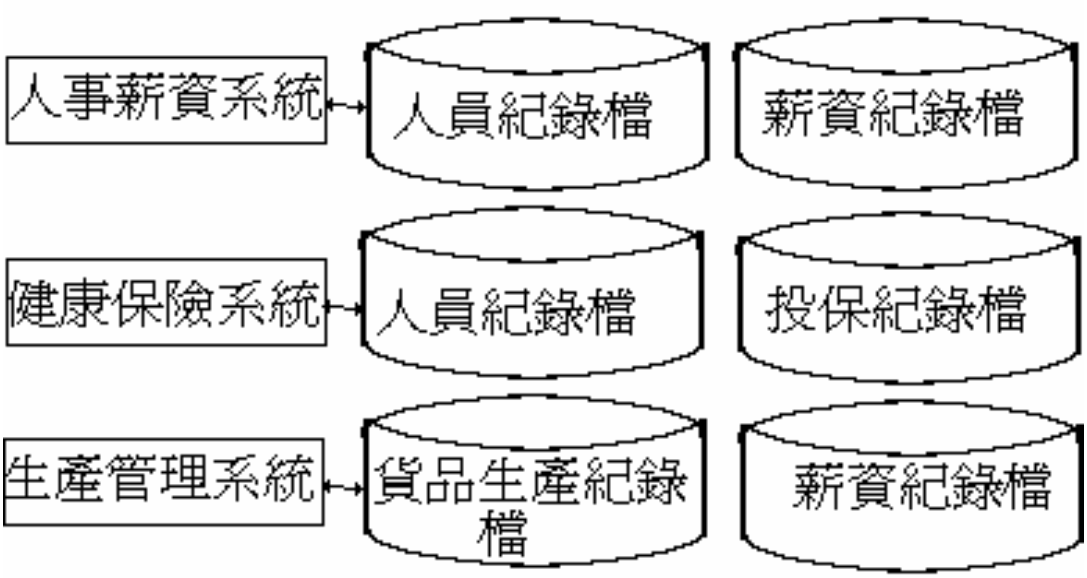
<資料庫系統與檔案系統>

資料必須透過許多步驟的處理之後才能轉變成為有關的資訊.長久以來電腦的主要角色就是將那些已經收集到的資料,就其特性其屬性來進行整合處理.在過去所有的資料,大多是以一種簡單的方式,儲存在電腦系統內的檔案之中,這些檔案最主要的特性就市可以讓各種應用程式直接存取其中內容.舉例來說,某一個企業將他所有的員工都以一個檔案來儲存時,這個企業就可以開發一套人事資料管理系統,來直接讀取增加及修改員工個人資料.

這種資料管理方式比較直接與簡單,可是對於整個軟體的生命週期來計算,這種以檔案為基礎的資料結構容易造成許許多多的不變與困難.因此,另外一種與資料庫管理系統來管理資料的方式,就漸漸取代已檔案系統為主的管理方法.

<檔案系統處理的方法>

在資料庫被廣泛使用之前,大多數的資料處理系統是直接針對檔案系統來執行,資料的格式與資料檔的結構都與資料處理系統中的流程何演算密不可分.就像是早期非常流行的 COBOL 程式語言,就是將算程式,資料格式及資料檔架構,相互結合在一起的一種程式語言,在當時,這種做法的好處是較簡單清楚,而且處理速度也不錯,也因此 20 到 30 年前的 COBOL,自今仍然無法被取代.當然,這種處理方法的優點,就是建立在程式與資料的高度相互結合之上,但是不幸的,這重也是它產生問題之所在.就以下圖來看,每一個不同的應用程式,如人事薪資系統,都會存在一個或一組的檔案,來儲存相關資料.如果不同的印用程式都個別使用不同的資料檔,首先就會產生資料重複的問題.



資料重複的問題

3.我們的學習內容

附件　瑣碎的文件與畫面

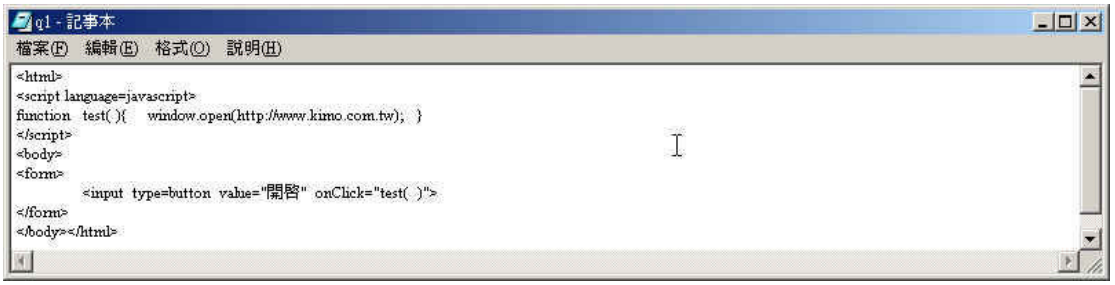


圖 A-1 我們對文件的學習之一，問題一。

圖 A-2 我們對文件的學習之一，問題二。

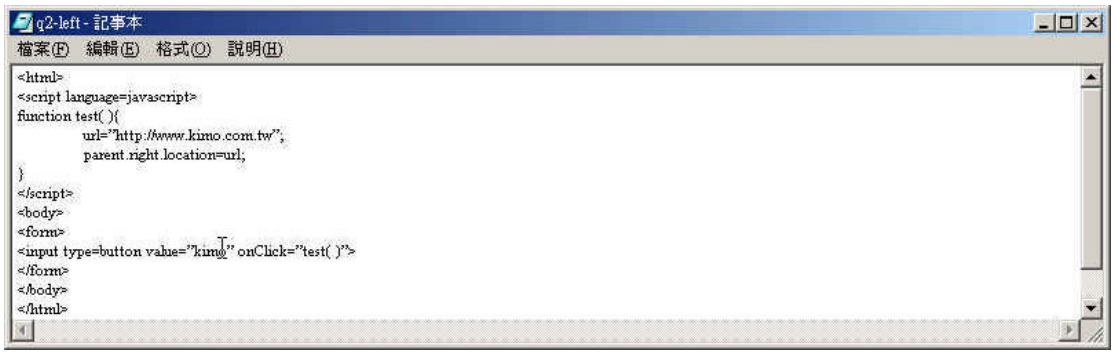


圖 A-3 我們對文件的學習之一，問題二。

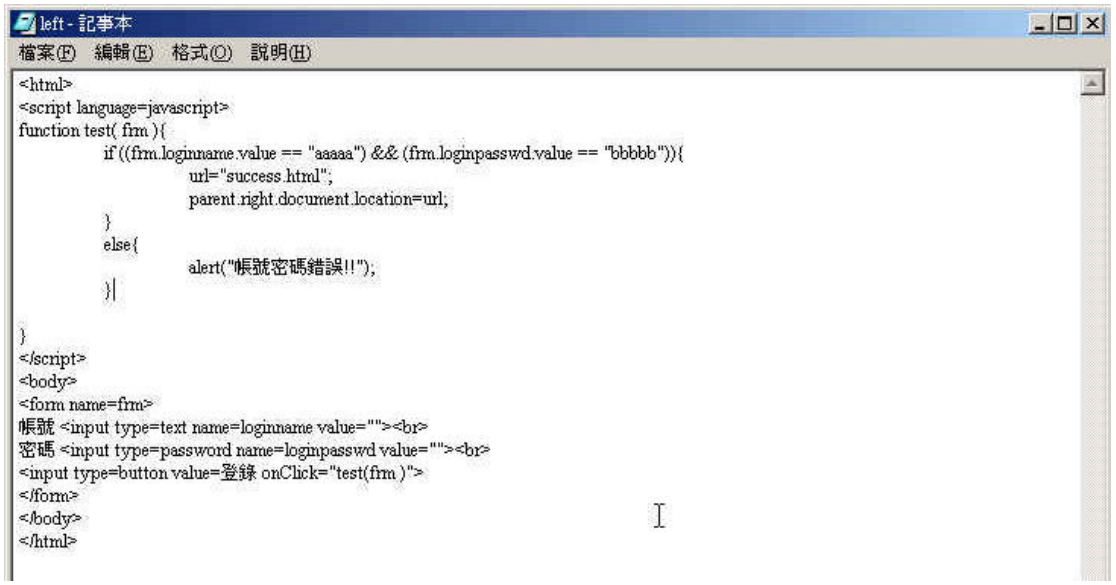


圖 A-4 我們對文件的學習之一，問題三。

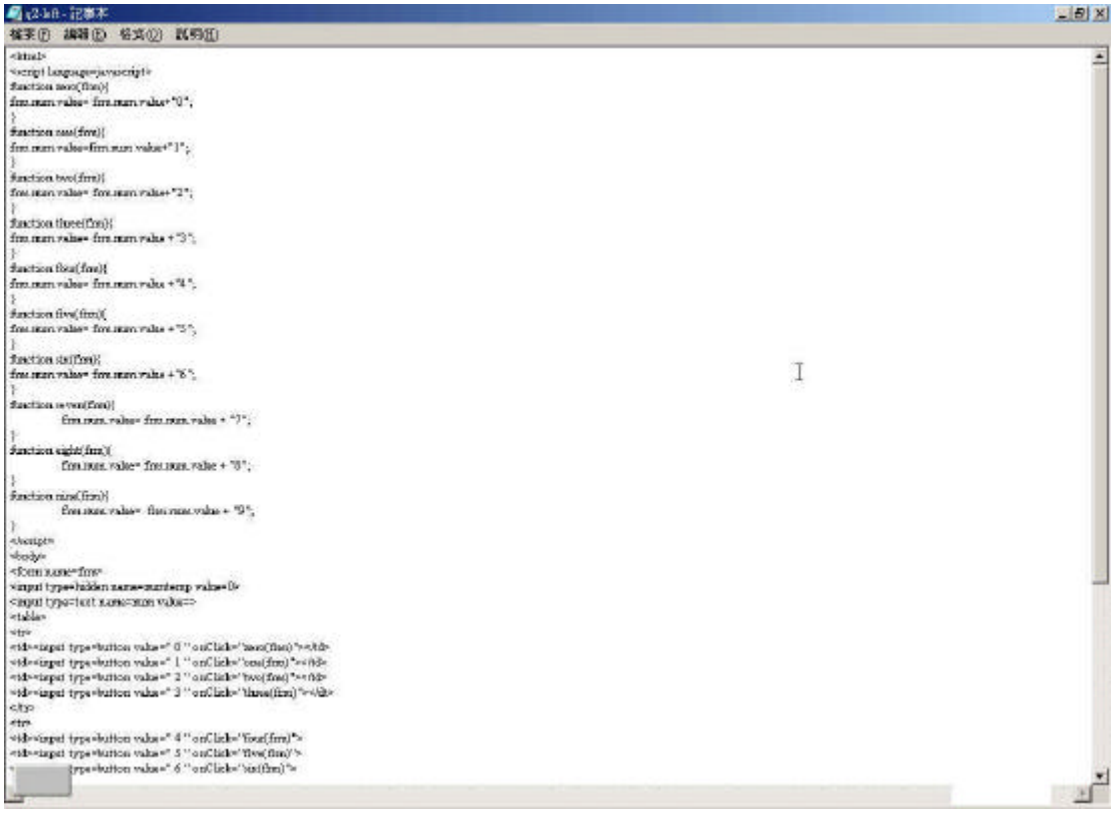


圖 A-6 我們對文件的學習之一，問題五。

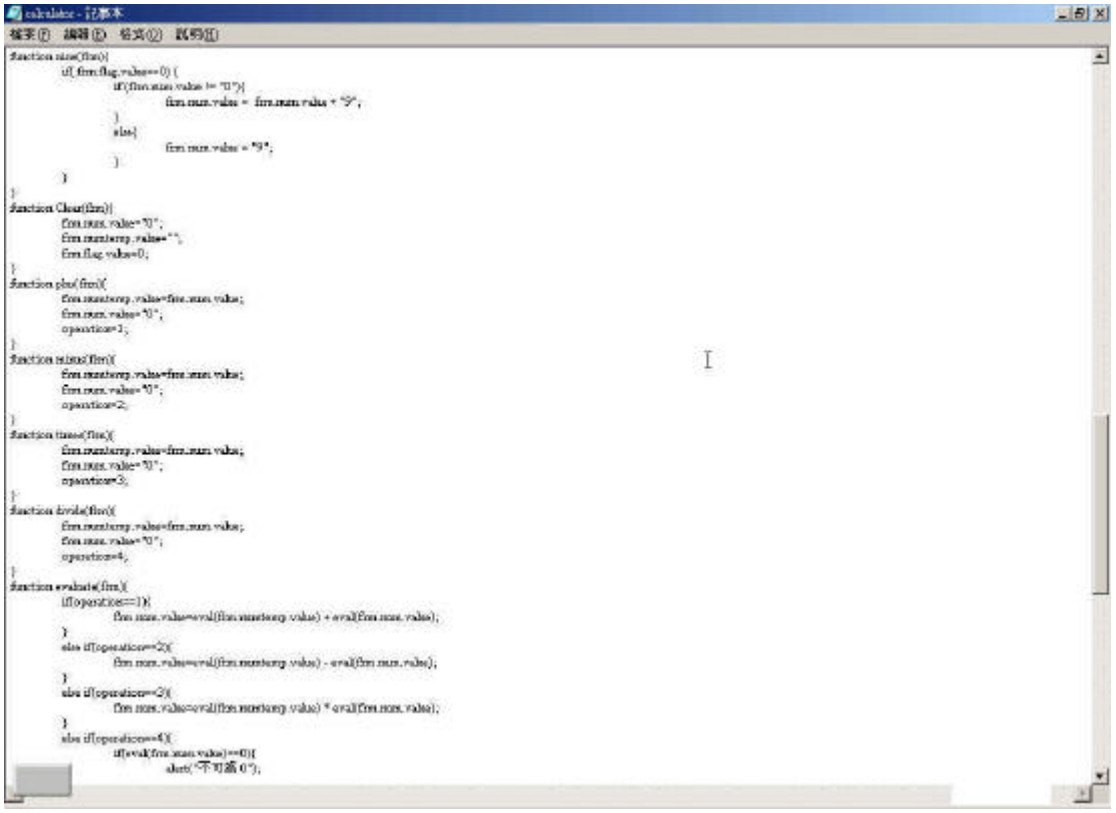


圖 A-7 我們對文件的學習之一，問題六。

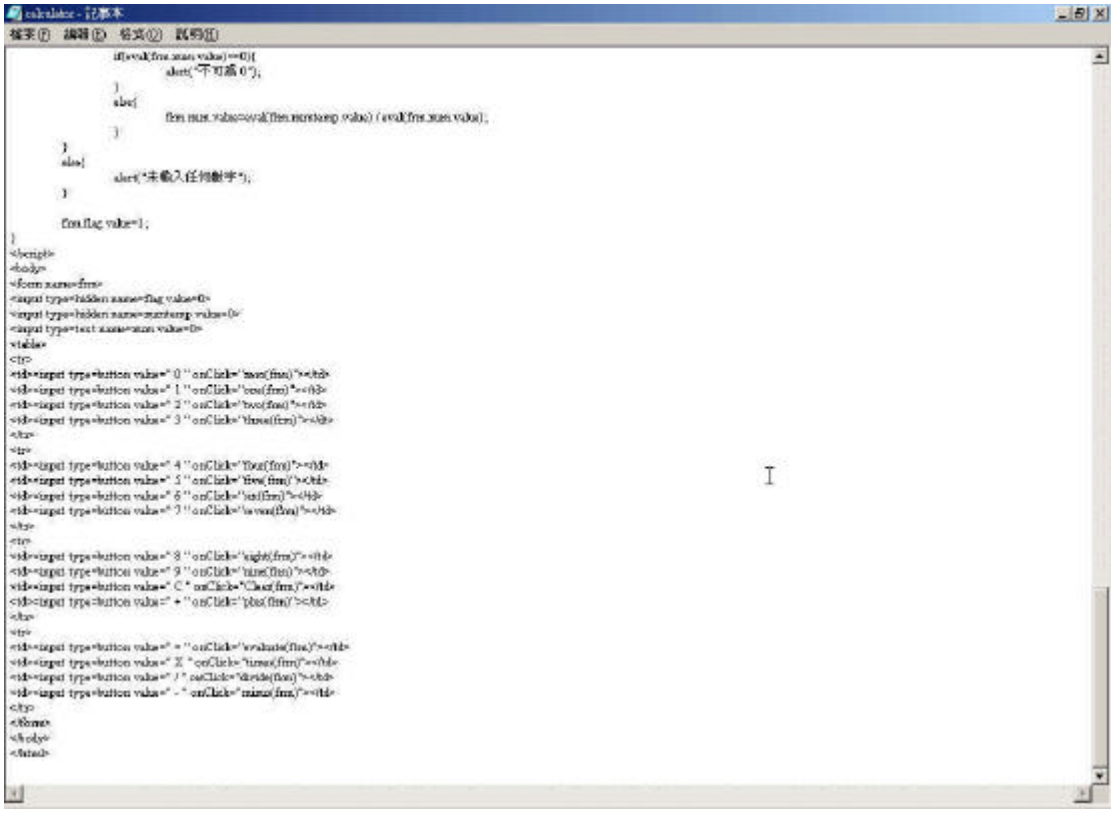


圖 A-8 我們對文件的學習之一，問題六。

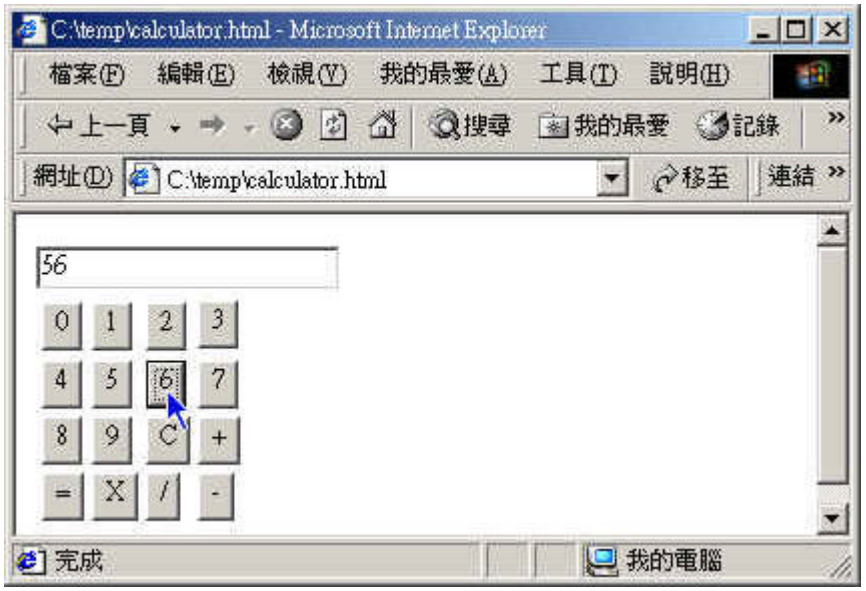


圖 A-9 我們對文件的學習之一，問題六。

q8-1 - 記事本

檔案(F) 編輯(E) 格式(O) 說明(H)

```
<html>
<head>
</head>
<script language="JavaScript">

var  netscape = (navigator.appName == "Netscape");
var  n4 = netscape && (parseInt(navigator.appVersion) >= 4);
var  explorer = (navigator.appName == "Microsoft Internet Explorer");
var  ie4 = explorer && (parseInt(navigator.appVersion) >= 4);

function shake(n) {
        if (n4 || ie4) {
                for (i = 10; i > 0; i--) {
                        for (j = n; j > 0; j--) {
                                self.moveBy(0,i);
                                self.moveBy(i,0);
                                self.moveBy(0,-i);
                                self.moveBy(-i,0);
                        }
                }
        }
}

</script>

<body>

<FORM>
<input type=button onClick="shake(5)" value="顫動">
</FORM>
</body>
</html>
```

圖 A-10 我們對文件的學習之一，問題八。

q8 - 記事本

檔案(F) 編輯(E) 格式(O) 說明(H)

```
<html>
<script language="JavaScript">
today=new Date();
var h = today.getHours();
var m = today.getMinutes();
var t = " " + (h) + "<BLINK>:</BLINK>" + (m) + "分";
function initArray(){
this.length=initArray.arguments.length
for(var i=0;i<this.length;i++)
this[i+1]=initArray.arguments[i] }
var d=new initArray("星期日", "星期一","星期二", "星期三","星期四", "星期五", "星期六");
document.write("民國",today.getYear(),"年",today.getMonth()+1,"月",today.getDate(),"日",d[today.getDay()+1]," ",t, "");
</script>
</html>
```

圖 A-11 我們對文件的學習之一，問題八。

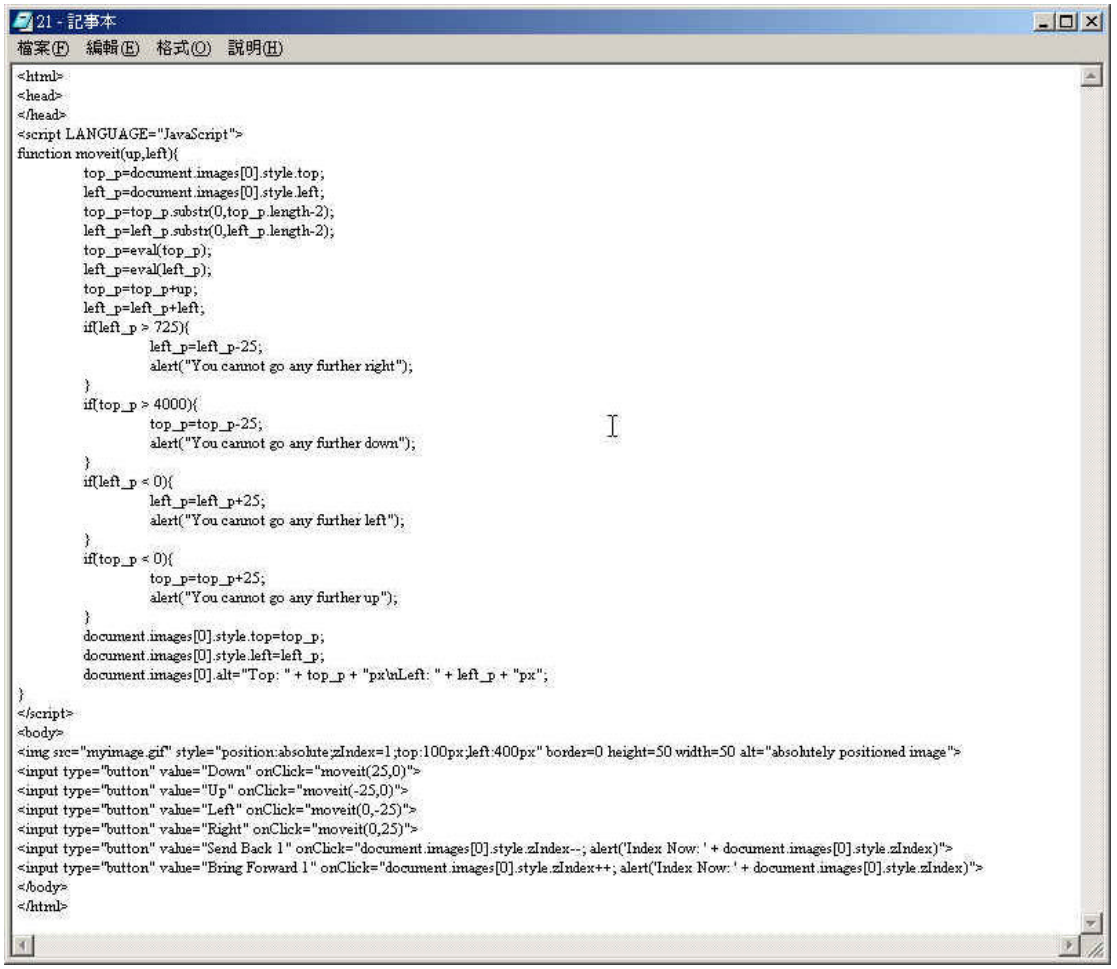


圖 A-12 **我們對文件的學習之一，問題九。**

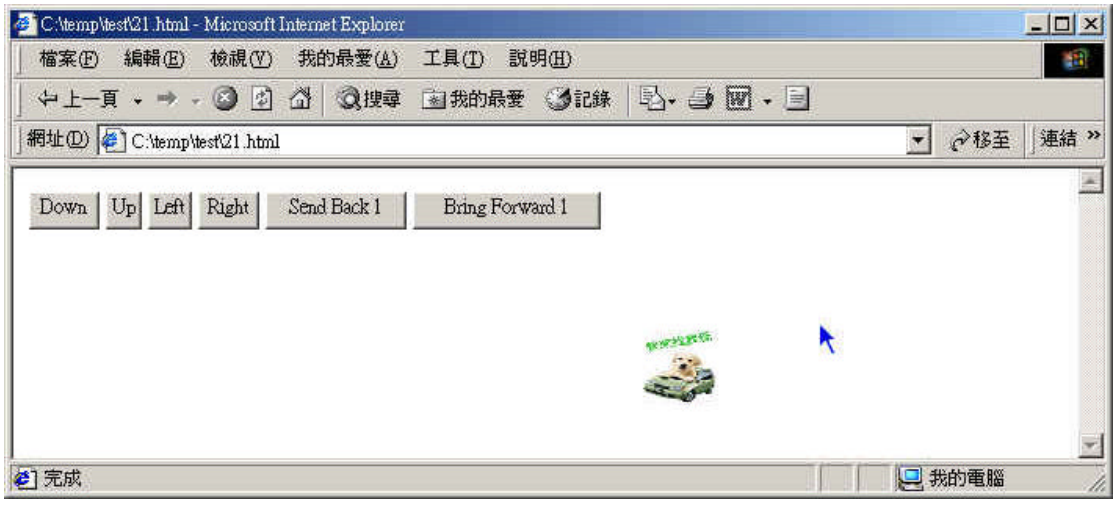


圖 A-13 **我們對文件的學習之一，問題九。**

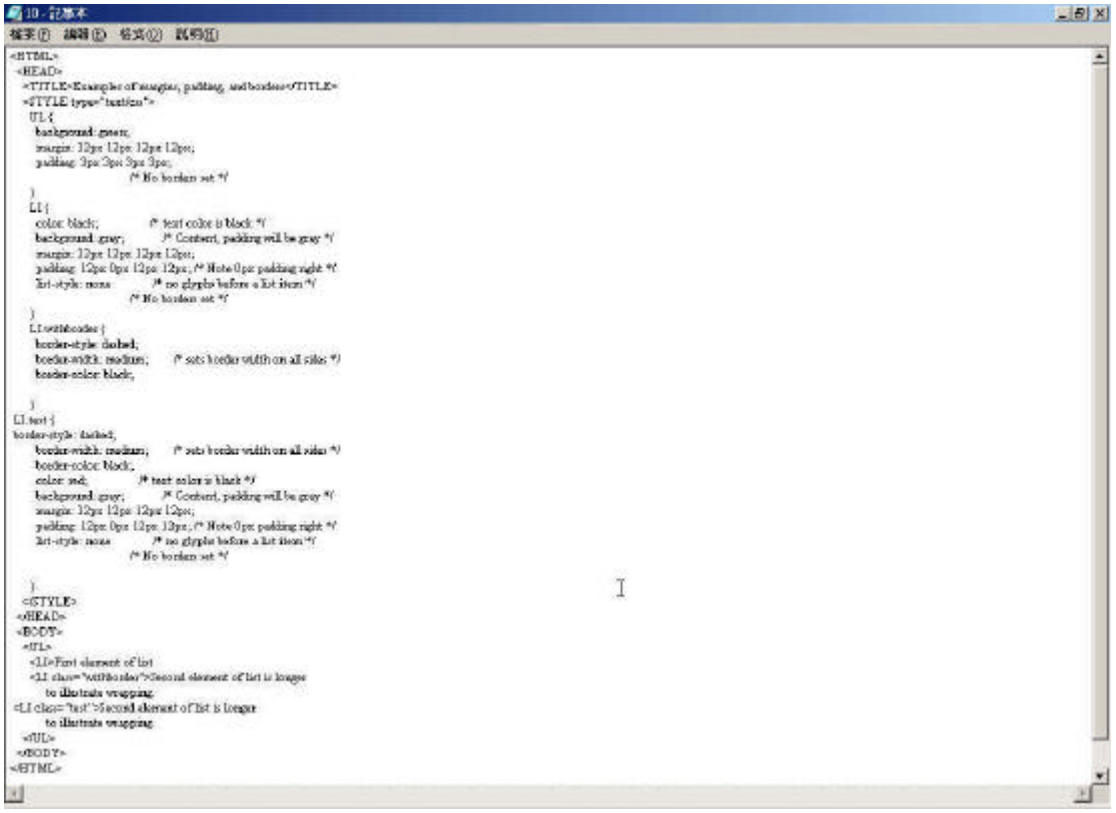


圖 A-14 我們對文件的學習之一，問題十。

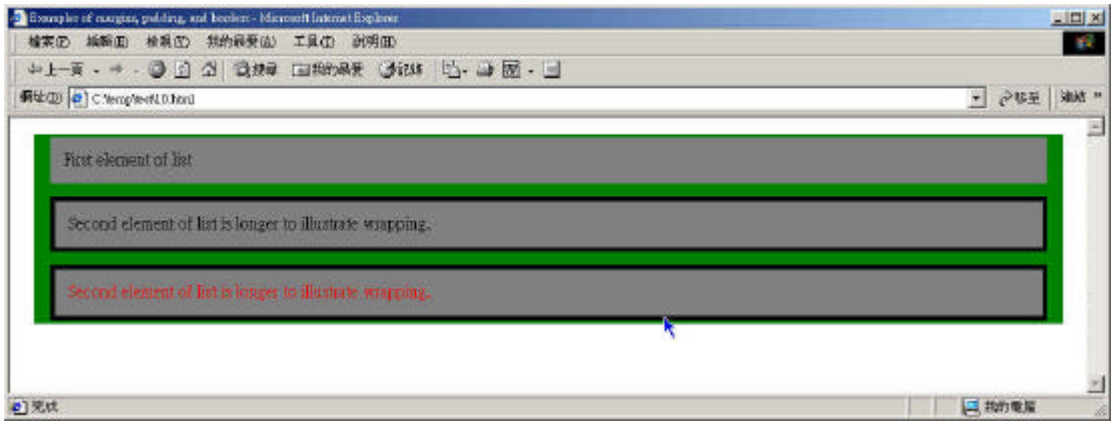


圖 A-15 我們對文件的學習之一，問題十。

我們的學習

以下以問題的形式，說明我們對動態文件設計技術的研究。

答：(問題一)

- **瀏覽視窗例如** IE 5.0 **與** Netscape 4.5 **解譯** HTML **文件，將** HTML **文件呈現於視窗中，它還具有執行** javascript **程式敘述的能力。在此** HTML **文件中加入下列段落**

```
<script language=javascript>
        function test(    ){          window.open("http://www.kimo.com.tw")；  }
</script>
```

是 JavaScript 草稿語言，它是動態網頁設計技術之一。

- <script> 與 </script>宣告此處為一個 javascript 程式區段， 與 HTML 段落區隔，其間也定義了一個函式，名字叫 test； 此函式包含一個敘述 window.open("http://www.kimo.com.tw") 。
- 當 test( ) 函式被呼叫時， JavaScript 的敘述 window.open， 會被瀏覽視窗執行， 而執行的結果是使瀏覽視窗開起一個新視窗， 就好像母親生出了一個嬰兒一樣。
- window.open(參數) 是 window 內建的 open 函式， window 執行 open， 會開啟另一個視窗。完整的 window.open 敘述應為 window.open(URL， windowname，windowfeature)；其中， URL 表示要載入的文件網址， windowname 是新視窗的名稱， windowfeature 是此新視窗的一些特徵， 如 toolbar 等。例如
  window("http://www.kimo.com.tw"，"kimo"，"toolbar=no，menubar=no，scrollbar=yes，width=400，height=200") 。
- 此 HTML 文件被載入瀏覽視窗， 瀏覽視窗看到此 javascript 區段之後， 它會知道此程式區段的含意， 但可能不會馬上執行。
- 再看視窗中的按鈕，.

```
<form>
      <input  type=button  value="開啟"  onClick="test(  )">
</form>
```

這個 <input> 定義了一個按鈕，它的行為被事件 onClick 驅動， 是希望它被按下之後， 瀏覽視窗就呼叫 test( ) 函式， 而瀏覽視窗早先已知 test( ) 函式的含意， 因此便表現了開啟新式窗的動作了。 這是我們常用的一項設計， 用一個按鈕，開啟另一個網頁。

答(問題二)

- main.html
  - HTML 的 <frameset> 是告訴瀏覽視窗將 window 規劃成若干 frame 視框的方法。
  - 此處 <frameset cols=”30%,70%”> 形成左視框與右視框。
  - 此處 <frame name=left src=left.html> <frame name=right src=right.html> 告訴瀏覽視窗，左視框名稱叫 left，載入 left.html；右視框名稱叫 right，載入 right.html。
- left.html
  - <input type=button value=”kimo” onClick=”test( )”> 是 HTML 文件在左視框中出現表單的按鈕物件之一。 而 onClick 是事件名稱之一，是瀏覽視窗定義任何按鈕接受

滑鼠按一下的事件名稱， 此寫法就是希望使用者用滑鼠向此按鈕按一下，引發 onClick 事件，onClick 事件驅動 test( ) 涵式(function)被瀏覽視窗執行。

- <script language=javascript> 至 </script> 之間為 javascript 語言片段，當 left.html 載入左視框時， 瀏覽視窗 window 會預知的程式敘述。
- right 是在 main.html 中所定義的右視框物件名稱； javascript 語言 parent.right.location 表示 right 右視框物件的 location 屬性， 而視框物件的 location 屬性就是指網址。因此 parent.right.location=url 就是指定 url 變數的值給右視框網址。
- 因此，透過左視框中按鈕可在右視框瀏覽 kimo 網站是本例的主題。

答：(第三題)

將第二題的左視窗框網頁改為一個登錄的表單，如圖二所示。

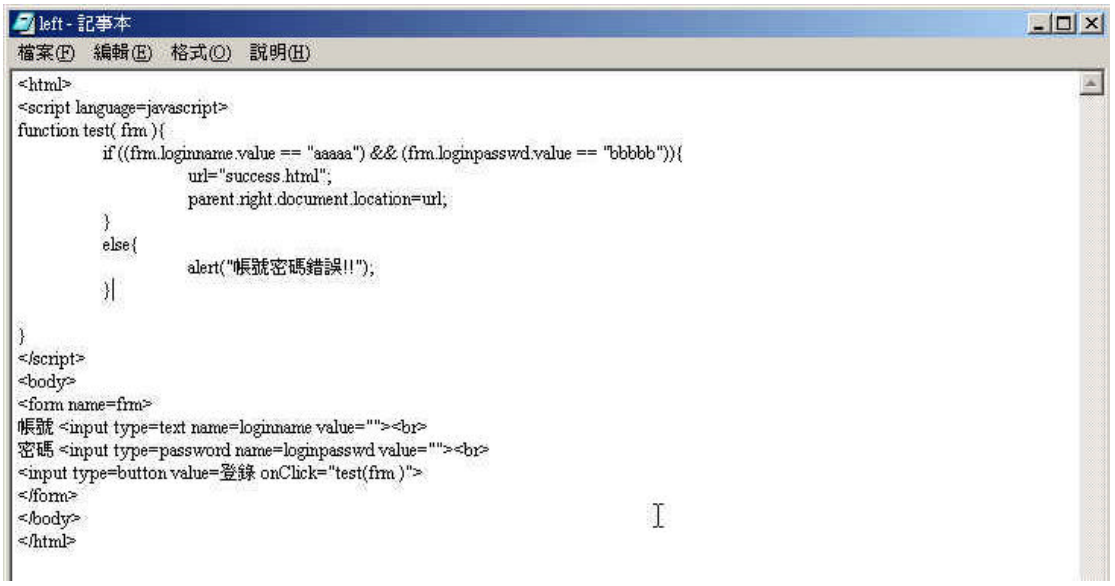


圖 2 左視窗框的文件 left.html。

圖 3 開啟主網頁文件 main.html，**見第二題。**

文件的設計上，定義出登錄的表單的名稱為 frm，**希望在** test(frm) **中判斷填入表單中的帳號密碼兩個欄位的值與內定的帳號與密碼作比對，** test(frm)**中的內定的帳號與密碼是** aaaaa **與** bbbbb。

注意一下， Javascript **程式的** if **敘述句型是作比對的最適當的句型，它是判斷不同值的邏輯句子，如本題的寫法，圖一所示。**

因此，藉此登錄機制的設計可控制右視窗框的網頁連結設定，此設定網頁連結句子如下：

```
url = “http://www.xxx.xxx.xxx”;
parent.right.location = url;
```

另外，瀏覽視窗執行 Javascript **程式的內建函式** alert(“**訊息文字**”) **會彈出一個訊息視窗，很好用，可善加利用。**

研究一下，內定的帳號與密碼的不安全問題？ 如何解決？

答：(第五題)

繼前題的觀念，表單中的輸入元件是 JavaScript **程式語言所定義的物件；這**

是非常重要的認識。

本題製作若干的按鈕物件各代表（模擬）計算器上的數字按鈕，而液晶面板式被表單的文字輸入物件來表示。

如 按鈕物件 <input type=button value=" 1 " onClick="one(frm)">

如 文字輸入物件 <input type=text name=num value=>

Script 中的函式是用來控制面板的顯示的機制，如

```
function one(frm){
    frm.num.value= frm.num.value + "1";
}
```

frm.num.value= frm.num.value + "1"; 是 frm.num.value 與 "1" 兩字串相連接的運算，例如 "123" + "abc" 會得到 "123abc" 的字串。

答：(問題六)

續第五題計算機介面的設計完成，本題要將按鍵的功能完整的設計。

模擬計算機的功用設計上技術歸納若干重點如下

A、 加減乘除的運算判斷落在等號（＝）按鈕的函式執行

operation 變數定義現在處理何種運算，各在 +、 - 、X 、/ 按鈕的函式內決定。

B、 flag 欄位定義是否為完成一次運算，若完成一次運算，結果顯示於面板之後，則不可按任何數字按鈕（frm.flag.value=1），清除（C）按鈕會清除它為 0，表可開始按任何數字按鈕。

C、 輸入一個數如 250 時，frm.num.value=”0”; frm.num.value = frm.num.value + "2"; frm.num.value = frm.num.value + "5"; frm.num.value = frm.num.value + "0"; 造成面板顯示會變成 0250 錯誤現象、而不是 250；所以，須處理此錯誤現象，如下列所示：

答：(第七題)

如前面數題目所舉的視窗按鈕動作的例子一樣，使用者用滑鼠對視窗按鈕物

件的一個 click 動作，基本上就是觸發一個 onClick 事件於視窗按鈕物件上；我們就可對按鈕定義該事件應該要執行的 javascript 敘述函式模組。本題中的 onMouseover 也是描述使用者用滑鼠滑過文件表格內的文字的動作，而 onMouseout 是描述使用者用滑鼠滑出文件表格內的文字的動作。

W3C 組織約從 1990 年起開始規範 HTML 文件的標準，至今已十餘年。期間有對 StyleSheet 的增訂，它們即是 CSS1、CSS2、與 CSS3。而如今像 Netscape 6.x 與 IE 5.x 以後的版本的瀏覽視窗都以具有解譯含有 StyleSheet 的 HTML 網頁文件的能力，它們被稱為 CSS Browser；因此，本題有必要對 W3C 的制定文件作些研究。

參考 www.w3.org 可得 W3C Core Sytle

```
<html>
<head>
<title>Document title</title>
<link rel="stylesheet" href="http://www.w3.org/StyleSheets/Core/Modernist" type="text/css">
</head>
```

定義：

```
<tr bgcolor="yellow" style="cursor:hand" onmouseout="this.style.backgroundColor='green'"
                    nmouseover="this.style.backgroundColor='#feb09d'">
```

stylesheet 的規範

style 用於指定某樣式，如 cursor:wait

javascript 的規範

this 指這個 <tr> 列

style 指樣式

backgroundColor 指背景色

html 的規範

bgcolor 指背景色

答：(第八題)

瀏覽視窗具有解譯執行 HTML 文件中的 Javascript 程式語言的能力，瀏覽

視窗的各種物件皆在 Javascript 程式語言中有定義；如 window 物件、navigator 物件、form 物件、button 物件、text 物件等等。

navigator 是定義現在使用瀏覽視窗的應用程式物件。

navigator.appName 是定義現在使用瀏覽視窗的名稱。

navigator.appVersion 是定義現在使用瀏覽視窗的版本。

self 是定義現在使用啟動的瀏覽視窗物件。

self.moveBy(x,y) 是 self 物件移動的行為以整個螢幕的現在位置(0,0)算起至(x,y)的位置。

setTimeout("A",B) 是設定 A 函式多久（即 B x 0.001 秒）要被瀏覽視窗重複執行一次。A 代表某一函式的名稱、B 代表一個時間。

var today = new Date( ) 是宣告一個時間物件變數 now，它的值等於 new Date( )；而 new 是 Javascript 程式語言定義的一個運算子，以物件導向程式設計的觀念來說，new 是建構子(creator)，產生一個物件實體； Date( )是 Javascript 程式語言定義的日期時間函式。

因此，now 是一個時間物件實體變數，當文件被載入瀏覽視窗時，即得到現在的日期時間。

Javascript 程式語言已對時間物件定義了它的物件行為方法，如 today.getHours( )、today.getMinutes( )、today.getSeconds( )、today.getYear( )、today.getMonth( )、today.getDay( )等等。

today.getHours( ) 取得 now 的「小時」資料。

today.getMinutes( ) 取得 now 的「分」資料。

today.getSeconds( ) 取得 now 的「秒」資料。

today.getYear( ) 取得 now 的「西元年」資料。

today.getMonth( ) 取得 now 的「西元月」資料。

today.getDay( ) 取得 now 的「西元日」資料。

第二章 開發成品的協助

系統的準備

<什麼是網際網路??>

在電腦發展的初期,電腦只不過是放滿一整個房間的"龐然大物",只有某些特定的人才會去使用電腦,而那時電腦也只是設計來作科學的運算而已.但再經過十多年的努力以後,電腦已經是放在每一個人的書桌上的多用途工作了.每一個人都知道電腦可以幫助我們撰寫文件,計算帳目,可以提高我們日常工作的效率,可以在電腦上玩遊戲,成為我們日常的休閒之一.但是進幾年來人們逐漸發現坐在一部電腦前面工作,娛樂已經不能滿足他們了,大家都期望把一部部的電腦串聯在一起,以提高工作的效率與娛樂的效果.所以電腦不再是一部部的存在;而是 2 部 3 部的一起運作.最簡單電腦連線的形式是透過每一部電腦上的硬體的連結埠利用連接線將兩部電腦連接在一起,然後這兩部電腦便可以利用連接埠將資料或訊息透過連接線傳到對方電腦的連接埠中,而對方只要從連接埠中將資料或訊息讀出既可達到傳訊的目的.這就是最簡單的電腦網路.再利用連接埠做為電腦網路的通訊設備中,最大的瓶頸便是連接埠的資料傳輸率太低以及這種連接型態是以連接兩部電腦作為基準的考量,所以要利用這種連接部將兩部以上的電腦連結在一起時便會大大的增加網路的成本,與網路軟體的複雜度.所以利用電腦上所額外加上的"網路卡"做為網路傳輸裝置的觀念便出現了.各個公司紛紛設計利用各種電器特性的網路卡與網路硬體連接方式,使得電腦網路的定義又邁向了新的一頁而後便出現了如:Token Ring Token Bus 與 Ethernet 的等大家耳熟能詳的網路系統.而在這些以網路卡,同軸電纜為主角的網路系統中有一些共同的特性,那就是傳輸速率高,網路架構單純,網路節點數不限等.而這種方式便構成了我門目前最主要也最普遍的電腦網路系統.

SQL Serve 資料庫管理與應用學習

<何謂 SQL Server>

簡單地說,SQL Server 是一個主程式(client/Server)架構應用統中所使用的關聯性資料庫管理系統(RDBMS),其中具有結合前端應用系統的資料庫應用系統架構,讓你的應用系統對於資料處理的能力,可以採用前端應用系統處理,或者是透過後端的 SQL Server 進行處理.

一般前端應用系統與 SQL Server 結合應用架構,可以區分如下面所列的幾個標準的架構元數:

前端一般存在於應用系統程式與介面,其中包含有使用著界面的顯示,與配合操作模式設定的 SQL 敘述.

由前端應用系統將宣告的 SQL 敘述採用如圖 1.1 所示的應用架構,將 SQL 敘述送至後端的 SQL Server 中進行執行.

後端 SQL Server 執行符合該資料庫伺服器所支援的 SQL 敘述之後, 可以產生結

果(Results).
這其中包含有 SQL Server 對資料庫完整性的管制作業,以及使用著端進入 SQL Server 的安全性管制作業等等,如此形成一個後端完整且具有安全性的關聯性資料庫伺服器管理作業.
<SQL Server 所使用的語言>
如前面所介紹的前端應用程式介面,可以是視覺化 Visual Basic. Visual FoxPro 等工具,或者是結合 web 伺服器的網站應用系統,透過瀏覽器瀏覽結果的應用,這其中不管是你透過視覺化工具,或是網站應用系統中的 ASP 動態網頁,其中所撰寫的存取 SQL Server 敘述,必須符合 SQL Server 所使用的 T-SQL.
所謂 T-SQL,實際上為 Transact-SQL,為一個交易的 SQL 的語言,其中關於 SQL 意義為 Structured Query Language,一般解釋為結構式查詢語言,SQL 使用 American National Standards Institute (簡稱 ANSI)於 1922 所公佈的標準,所以我們可以說 SQL Server 的 SQL Server 的 SQL 語言符合 ANSI92 之標準.
因為各廠家所支援的 SQL 語言,均有其延伸的功能,而 SQL Server 所使用的 SQL 語言不外有 Microsoft 所加強之功能,如函數等應用,故稱呼為 T-SQL.
當前端設定好一個存取後端 SQL 資料庫的遠端資料集之後,實際上對應的即是 SQL Server 的 T-SQL 敘述.
安裝 SQL Server 7.0
在進行 SQL Server 概觀嘹解之後,接下來我們可以進行 SQL Server7.0 的安裝作業,但是當你欲進行 SQL Servre7.0 安裝之前,你扔得進依步瞭解瞭解愈進行安裝的平台與 SQL Server7.0 安裝時的配置需求等資訊,逐一說明如下:
<系統需求>
首先介紹關於安裝 SQL Server7.0 的硬體配置需求,說明如下:
電腦硬體
使用 Intel 相容性的電腦
DEC Alpha 相容性電腦
記憶體需求:
你的電腦記憶體配置最少為 32MB RAM,當然這是最少的需求,建議你扔擴充你的記憶體需求.
另外安裝 SQL Server Enterprise 版本時,最少需求的記憶體為 64MB RAM.
硬體空間需求
SQL Server 安裝最少需求空間為 70MB,快數安裝時需要 170MB,另外關於管理工具安裝時需要 90MB 左右.
若你又安裝了 SQL Server7.0 中所提共的 OLAP Server,這時又需要 50MB 左右的空間.
檔案格式
你可以交硬碟劃分為:
FAT 格式

NTFS 格式
建議你使用 NTFS 格式,具有 NT Server C2 Class 安全性.
<作業系統需求>
前面介紹過 SQL Server7.0 可以安裝在 windows 95/98’NT Workstation 與 NT Server 不同平台中,這時候若你欲安裝在相對平台中,其中附加的作業需求為何?說明如下:
SQL Server Enterprise 版本
SQL Server7.0 在包裝上區分為 Enterprise 與 Standard 兩個版本,在這兩個版本中均另外支援 Desktop 安裝方式,若你欲安裝的為 SQL Server7.0 Enterprise 版本,這時候你所需要的 NT Server 平台為:
NT Server4.0 Enterprise 版本
Service Pack4
SQL Server7.0 Standard 版本
若你欲安裝的為 SQL Servetr7.0 Standard 版本時,你需要的平台環境為:
NT Server4.0 或 NT Server Enterprise 版本
Service Pack4
另外請注意你必須安裝相對語系的 NT Service Pack4.
SQL Server7.0 Desktop 版本
所謂 SQL Server7.0 Desktop 版本,並非是一個獨立包裝的版本,這是一個版本分別存在於 SQL Server7.0 Enterprise 與 SQL Server7.0 Standard 版本中,你可以再這一薛版本中安裝 SQL Server7.0 Desktop 版本,其所需要的平台為:
Windows 95/98
NT Workstation 4.0 加上 Service Pack 4
NT Serever4.0 加上 Service Pack 4
NT server 4.0 企業版加上 Service 4
<平台網路環境>
一般你可以使用 Windows 預設的網路環境,SQL Server 支援 Web 伺服器進行網路存取,或透過網際網路進行遠端管理,你可以在你安裝 SQL Server 的平台中設定好 TCP/IP 通訊協定,其餘採用標準預設即可.
<瞭解你的 SQL Server 組態設定>
當我們透過 Enterprise Manager 註冊一個 SQL Server 進行管理之後,也就是偶而你會常使用這樣的工具進行資料庫管理與維護作業,這時候你應該了結你所註冊的的 SQL Server 預設的組態為何?如何瞭解一個 SQL Server 組態設定?其操作程式序說明如下:
啟動 SQL Server 組態設定訊息
首先你可以透過 Enterprise Manager 選擇一個註冊的 SQL Server,並且順利連接上該 SQL Server 也就是圖示附上了紅色取線,表示你已經連接上該 SQL Server.

接下來執行 SQL Server 項目右鑑功能表列中的 Properties 項目,即可啟動 SQL Server Properties 對話盒,顯示該 SQL Server 設定之組態,顯示結果.
這一個屬性視窗具有多個標籤頁,可以進一步瞭解整個 SQL Sever 預設的屬性狀態,其屬性說明如下:
一般屬性內容
第一個標籤頁內容為 General 標籤頁中,顯示了關於 SQL Sever 所安裝的版本,平台版本,以及所安裝的 Cord Page 等訊息.
記憶體管理模式
第二個標籤業為 Memory 標籤頁,在這一個標籤頁中顯示了 SQL Server 採用動態館鋰記憶體方式,透過 SQL Server 自動配置所需要的記憶體,當然其中記憶體最大限制牽涉到該機器中的記憶體大小.
處理器資訊
第三個標籤業為 Processor,表示你可以設定在一個多重處理器架構下,SQL Server 可以使用的處理器為哪些?只要是套用在 SMP 架構下,你可以適當調整 SQL Server 所能使用的處理器為哪些/同時瞭解其中最大的執行緒.
安全性登錄方式
第四個標籤頁為 Security,只要是關於登錄 SQL Server 身分認證方式,以及啟動 SQL Server Services 帳號設定,該對話盒預設採用登錄 SQL Server 身分認證方式微 SQL Server Standard 與整合 NT 網域帳號兩種方式.
連接設定方式
第五個標籤頁為 Connection,主要設定於 SQL Server 允許的連接數量,以及遠端市否採用 RPC 方式進行連接.
伺服器設定第六個標籤頁為 Server Setting 設定,只要是關於 SQL Server 的設定,其中包含有關於預設的語系,以及是否支援 SQL server 7.0 新增的 Trigger 巢狀觸發架構,另外是否設定 SQL Mail 連結至 Mail Server 中.
資料庫設定
最後一個標籤頁為 Database Setting ,即是資料庫設定,其中包含有重建索引的特性,以及備份等待的方式.

- 什麼是 PHP

PHP 簡介
如果說 PHP 是 UNIX 系統上的 ASP, 那麼大家都應該知道 PHP 大概是什麼東西了。比起 ASP 的高知名度, PHP 似乎不太受人注意。其實國外的網頁使用 PHP 來建構網站已經有很長的一段時間了，最近 PHP 也開始在國內萌芽，PHP 也漸漸被注意，這是非常令人欣慰的一件事。

大家可能對 PHP 的全名比較感興趣，第一次接觸 PHP 時，PHP 當時還是叫做 Personal Home Page 或 Personal Homepage Program，至於現在的官方全名則

是 Hypertext Preprocessor。

PHP 是一種 server-side HTML-embedded 的 script 語言。這樣就很清楚了，PHP 是內嵌於 HTML 檔案裡的 script 語言。PHP 的功能與能力與一般使用 Perl 或 C 寫成的 CGI 沒二樣，但重要不同的是：一般 CGI 程式必須自行輸出或處理 HTML，而 PHP 是穿插在 HTML 裡的程式。基於這個重要的相異點，PHP 在 HTML 的處理上就頗為輕鬆。

PHP 與 Perl 處理 HTML 的比較

利用 Perl 寫成的 CGI 程式：

```
#!/usr/bin/perl

print "Content-type: text/html\n\n";

$name = "Paul";
print "<html>";
print "<head>";
print "<title></title>";
print "</head>";
print "<body>";
print "<p>Hi! I'm $name</p>";
print "</body>";
print "</html>";
```

再看同樣的輸出利用 PHP 要如何撰寫：

```
<html>
<head>
<title></title>
<?php $name = 'Paul' ?>
</head>
<body>
<p><?php echo "Hi! I'm $name" ?></p>
</body>
</html>
```

是不是變的簡單方便多了。

- PHP 的用途

PHP 主要用來設計 Web Applications。二個最主要的功能為：
連接後端資料庫
處理 FORM
最常見的應用是 PHP + MySQL，PHP 利用 MySQL API 配合 MySQL 資料庫管理系統可以設計出常見的網頁應用程式：

會員系統
線上論壇
電子報系統
投票程式
計數器
也有人利用 PHP 發展較大型的應用程式，例如 PHP 的 CRM (客戶關係管理) 軟體，其它較有名的像是 MySQL 管理系統 - phpMyAdmin 等。

除此之外，PHP 也可以寫 script (命令稿程式)，藉著 PHP 與 MySQL 良好的 API 界面，也有人利用 PHP 設計 script 來管理、維護資料庫。

- PHP 的優點

利用 PHP 寫 Web Application 有很多優點，在這裡 Jollen 不將 PHP 與其它用途相同的語言 (如: ASP) 做比較，但是我們要知道，PHP 到底有什麼過人之處，而利用 PHP 寫網頁應用程式時，又有那些迷人的優點。
內嵌於 HTML

這點當然是 PHP 或 ASP 的優點，比起 Perl 等語言，PHP 可以減少相當多處理 HTML 的時間，我們只要在 HTML 的適當位置寫程式即可。如果是利用樣板 (template) 來發展程式，那麼彈性更大。

PHP 是 CGI

PHP 不但是設計 CGI 程式的好工具，而且 PHP 可以說是動力加強版的 CGI 語言。例如，在處理 FORM 的輸入資料時，PHP 就有比別人更容易的方式。

PHP 易學易用

PHP 語法類似 C 語言，因此 PHP 本身並不難學，也相當容易撰寫。

PHP 網路資源豐富

PHP 在美國是相當紅的，而且相關網路資源、文件、電子書、免費的函式庫、免費的應用程式、免費的工具是相當多的。

Zend 噴射引擎

Zend 系列對 PHP 全力的支援，使得 PHP 如虎添翼，Zend 讓 PHP 強、還要更強，例如 PHP 4 全新採用 Zend 的 parser 引擎，大幅提升直譯 (interpreter) 速度，其它如 Zend Cache、Zend Compiler 更是神奇。

API 支援完整

這是讓 PHP 容易使用的原因之一，想想看，利用 Perl 如何存取 Sybase 資料庫管理系統？PHP 支援了多種資料庫管理系統的 API，例如：MySQL、Oracle、Sybase、Postgreqsl...等。

OO 與 PEAR 架構相當好用

PEAR 是利用 PHP 提供的 class 所設計的 class library，PEAR 讓 PHP 的程式碼更具重用性。PEAR 存放許多常用的 class library，例如資料庫的 class、處理 HTML 的 class 等等，而且還持續地增加當中。藉著簡單 class 的支援，嘿！這相當的好用。

PHP 與 Apache 緊密結合

PHP 支援 apache 的 DSO (Dynamic Shared Object) 安裝方式，與 apache 這個全球佔有率第一的 web server 有相當好的相容度。

支援 Session 與 Cookie

在 Web Application 的應用上，沒有 session 與 cookie 簡值無法生存，PHP4 內建對 cookie 的支援，使得 PHP 已全面支援 session 與 cookie 機制。甚致在沒有 session 的環境下也能使用 cookie。

- Hello, world!

簡單的 PHP 程式長相

一開始，Jollen 以一個 PHP 程式做為開頭。這是一份相單簡單的 PHP 程式：

```
<html>

<head>
<meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=big5">
<meta NAME="GENERATOR" CONTENT="Microsoft FrontPage 3.0">
<title>PHP3 Example.</title>
</head>
<body BGCOLOR="#FFFFFF">
<?php
echo "<p>Hello!</p><br>";
echo "<p>World...</p><br>";
?>
</body>
</html>
```

當這份 PHP 程式被執行後，會輸出底下的 HTML 給瀏覽器：

```
<html>

<head>
<meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=big5">
<meta NAME="GENERATOR" CONTENT="Microsoft FrontPage 3.0">
<title>PHP3 Example.</title>
</head>

<body BGCOLOR="#FFFFFF">

<p>Hello!</p><br>
<p>World...</p><br>
</body>
</html>
```

嗯...是的，沒錯！看起來就像是一個可以輕易做出來的網頁一樣，事實上呢，

這個網頁的前身就是用 Microsoft FrontPage 做出來的，看到了吧！

第一個程式藍色的部份就是 PHP 的程式區塊，PHP 的程式區塊利用 <?php ... ?> 圍起來，第二個輸出後的網頁藍色的部份就是 PHP 程式執行後的結果。也是使用者最後會接收到的檔案內容。

利用 PHP 的目的就是為了輸出 HTML 格式的檔案，只不過，用程式來跑可以讓網頁有「動態」效果，而利用編輯程式做出來的網頁是冰冰冷冷的！

請仔細比對原來的 PHP 程式被置換後的內容，PHP 程式執行後，只有「最終輸出的內容」才會呈現在畫面上，所以，當我們需要一些動態的內容時，就必須用 PHP 來設計網頁。這樣的情形最常見的像是顯示訪客的 IP 位址，因為 IP 位址的資訊無法事先利用編輯程式設計，也就是說，必須「動態」地顯示。

PHP 與 HTML 的關聯

接下來，我們又在這個簡單的程式裡發現一件重要的事。PHP 是用來輸出網頁內容的，而網頁不就是 HTML 寫成的嗎？是的，you got it！

這個重要的資訊告訴我們，PHP 被夾雜在 HTML 檔案裡，只有 PHP 的程式區塊會被執行，而執行後的結果，輸出時，必須要配合整個網頁所使用的 HTML 語法，輸出文字或圖形時，也要使用 HTML 的語法。

所以說：學好 PHP 前，要先把 HTML 搞定。

嗯，到這裡我想大家對 PHP 的長相已經有最初步的認識了，如果您想知道，要如何安裝並執行 PHP，那麼，請繼續看下去 :)

- 如何安裝 PHP

底下利用原始碼安裝。為什麼要用原始碼安裝呢？這是因為 Linux distribution 的版本太多，而利用原始碼來安裝則是最根本的方法，同時在編譯設定 (configure) 時也可以自己加上額外的參數。
安裝 Apache

下載 apache：

http://httpd.apache.org/dist/

這裡以 Apache 1.3.14 來說明安裝的程序，目前最新版本為 1.3.19。

安裝步驟：

下載 apache_1.3.14.tar.gz，將檔案放至適當目錄即可，筆者習慣是存到 /usr/local/src 目錄下。

解開檔案：

```
linux# tar zxvf apache_1.3.14.tar.gz
```

進到 apache 的目錄下，執行編譯檔案的設定：

```
linux# cd apache_1.3.14
linux# ./configure --prefix=/usr/local/apache --enable-module=so
```

加上 --prefix 參數表示我們要自定 Apache 的安裝目錄，
--prefix=/usr/local/apache 表示編譯完成的檔案要安裝在 /usr/local 目錄下。

加上 --enable-module 表示我們要額外編譯 module (模組) 進來，
--enable-module=so 表示要連同 mod_so.c 這個 module 一起編譯。因為會以 DSO (Dynamic Shared Object) 的方式安裝 PHP，所以利用原始碼安裝的話必須加上 --enable-module=so 的參數編譯 mod_so.c，才能使用 DSO。

進行編譯的工作：

```
linux# make
```

直接打 make 即可。編譯需要一點時間，請耐心等待，編譯完成後會回到提示字串下，然後再進行下一個步驟。

安裝 Apache：

```
linux# make install
```

執行 make install 即可將 Apache 安裝至 /usr/local 目錄下。

安裝 PHP

下載 PHP：

http://www.php.net/downloads.php

這裡以 PHP 4.0.2 的版本來做解說。

解開原始碼：

```
linux# tar zxvf php-4.0.2.tar.gz
```

設定編譯檔案：

```
linux# cd php-4.0.2
linux# ./configure --with-mysql --with-apxs=/usr/local/apache/bin/apxs
--prefix=/usr/local/php
```

參數用途：
--with-mysql： 表示要編譯 MySQL 的 API，這樣 PHP 才能存取 MySQL 的資料庫。
--with-apxs=/usr/local/apache/bin/apxs：還記得我們在編譯 Apache 時加上的 --enable-module=so 參數嗎，為的就是要讓 Apache 支援 DSO。加上 --with-apxs 參數表示要將 PHP 編譯成 DSO module 使用，
/usr/local/apache/bin/apxs 則是 Apache 的 apxs 檔案位置。
--prefix=/usr/local/php：將 PHP 安裝到 /usr/local/php 目錄下
編譯程式：

```
linux# make
```

安裝 PHP：

```
linux# make install
```

執行 make install 後即可安裝 PHP。

複製 php.ini：

linux# cp php.ini-dist /usr/local/php/lib/php.ini

將原始碼目錄下的 php.ini-dist 檔案複製到 --prefix 指定目錄下的 lib/，並更名為 php.ini。這個檔案就是 PHP4 的組態設定檔。

重新啟動 Apache Server：

linux# /usr/local/apache/bin/apachectl restart

安裝完成後，要記得重新啟動 Apache Server，PHP4 才能順利運作。如果沒有出現錯誤訊息，表示 PHP4 已經成功的以 DSO 的方式與 Apache 一起運作了。

設定 httpd.conf for PHP

在 httpd.conf 裡找到底下的設定項目，確定這些設定都有寫到 httpd.conf 檔案裡：

```
LoadModule php4_module libexec/libphp4.so
DirectoryIndex index.html index.php index.htm index.php3
AddType application/x-httpd-php .php .php3
AddType application/x-httpd-php-source .phps
```

如果 httpd.conf 沒有以上的設定項目，請自行增加這四行設定並做修改。為了能執行 PHP3 的程式，粗體字的地方是筆者額外加入的設定參數，也請讀者修改原來的設定，才能執行 .php3 的檔案。

在 DirectoryIndex 設定項目裡，我們加進了 index.php 的預設網頁名稱。習慣上，PHP4 的檔案都會存成 .php 的副檔名，而 PHP3 的檔案則是存成 .php3。如果要讓 PHP3 或 PHP4 不直接執行 PHP 的程式碼，而是列出檔案的原始程式碼，則要將檔案存成 .phps (PHP Source) 的副檔名，這就是最後一行設定的用途。

- PHP 入門觀念

PHP4 程式碼要寫在那裡
PHP4 的程式碼的寫法：

```
<?php
```

```
echo "Hello! World!";
?>
```

「<?php」 代表的是 PHP 程式碼的開頭，「?>」代表的是程式碼的結束，我們就將 PHP4 的程式碼寫在這兩個標籤之間。

範例：

```
<html>
<head>
<meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=big5">
<title>PHP Example.</title>
</head>
<body BGCOLOR="#FFFFFF">
<?php
echo "<p>Hello!</p><br>";
echo "<p>World...</p><br>";
?>
</body>
</html>
```

PHP 敘述的寫法

以 echo 語法為例，echo 是 PHP4 的語法，用來做輸出。而整個 echo 語法的撰寫則是一行完整的敘述，敘述結束時必須以分號做結尾，例如本文的範例：

```
<?
echo "Trust me!<br>";
echo "You can make it.<br>"
?>
```

敘述是容許斷行的，例如底下的例子：

```
<?
echo "Trust me!<br>
You can make it.<br>"
?>
```

PHP 與 HTML 的換行方式

**PHP 程式碼本文利用 Enter 做的換行動作並不等於在出現在瀏覽器畫面的換行，瀏覽器輸出畫面的換行必須使用 HTML 語法中的
 標籤：**

```
<?
echo "Trust me!<br>";
echo "You can make it.<br>"
?>
```

- **PHP 基本語法**

變數的寫法
PHP 的變數都是以 $ (dollar sign) 開頭，並且變數名稱有大小寫之分，例如：

```
$name
$Name
```

是兩個不同的變數。請看底下的例子：

```
<?php
$a = 5;
$a = $a*3-2;
echo "A = $a";
?>
```

輸出結果為：

```
A = 13
```

註解的寫法

PHP 的註解有二種形式：

1. 到行尾的註解：

```
$a = 5;         // assign 5 to $a
```

表示 「//」 之後到該行的結束均為註解。

2. 註解區：

```
/*
Variables &
Assignment
*/

$a = 5; /* assign 5 to $a */
$b = $a; /* assign $a to $b */
```

表示 /* 與 */ 之間均為註解。

if 敘述

if 敘述的格式：

```
if (EXPRESSION) {
   statement;
}
```

其義意為，當 EXPRESSION 為 true 時，則執行 statement。舉個簡單的例子：

```
if ($name == 'Jollen') {
   echo "Hi! Jollen";
}
```

當變數 $name 的值等於 Jollen 這個字串時，則顯示 Hi! Jollen 字串。當 statement 只有一行敘述時，可以省略大括弧。所以本例也可以寫成：

```
if ($name == 'Jollen')
   echo "Hi! Jollen";
```

搭配 else

加上 else 可以做「如果. . . 則. . . 否則. . . 」的邏輯判斷：

```
if ($name == "Jollen") {
   echo "Man!";
else {
   echo "Woman!";
}
```

再加上 elseif

elseif 的語法也是和 if 配合使用，請看底下的範例：

```
if ($name == "Jollen") {
   echo "Hi! Jollen.";
} elseif ($name == "Yii") {
   echo "Hi! Yii.";
} else {
   echo "Who are you?";
}
```

使用 elseif 做多個判斷時，如果判斷條件都不成立，那麼就會執行 else 裡的敘述。

while 敘述

while 是一種迴圈的敘述，語法如下：

```
while (EXPRESSION) {
    statements;
}
```

while 先判斷 EXPRESSION，如果 EXPRESSION 為 true，則執行 while 裡的述說：

```
$a = 1;
$sum = 0;
while ($a <= 10) {
   $sum = $sum+$a;
   $a = $a+1;
}
```

最後 $sum 的值為 55，即 1+2+3+4+5+6+7+8+9+10 的結果。

do...while 敘述

do...while 敘述也是迴圈敘述，但是 do...while 與 while 不同的地方在於，do...while 一定會先執行 do 裡的敘述一次，因為 while 是先去判斷 EXPRESSION 是否成立，而 do...while 則是先執行一次後才去判斷 EXPRESSION。

do...while 的語法：

```
do {
    statements;
} while (EXPRESSION);
```

範例：

```
$a = 1;
$sum = 0;
do {
   $sum = $sum+$a;
   $a = $a+1;
} while ($a <= 10);
```

最後 $sum 的值一樣是 45。

for 敘述

for 迴圈敘述是一種有條件式的迴圈語法，for 可以指定迴圈開始與結束的條作，因此可以限定迴圈的次數。for 語法：

```
for (EXPRESSION1; EXPRESSION2; EXPRESSION3) {
    statements;
}
```

其中 EXPRESSION1 為初始條件，EXPRESSION2 為終止條件，EXPRESSION3 為迴圈結束後所要執行的 statement。

範例：

```
for ($i = 0; $i <= 10 ; $i++) {
   echo "$i<br>";
}
```

執行時，$i = 0 表示 $i 的初始值為 0；$i <= 10 表示當 $i <= 10 時，迴圈繼續執行；$i++ 表示每次執行一次迴圈裡的敘述後 $i 的值加一。

$i++ 的寫法等於 $i = $i+1。

這裡有一個重要的觀念要說明：EXPRESSION3 是在迴圈結束後才被執行，例如：

```
for ($i = 0; $i <= 10; $i++) {
...
}
```

我們來看這個迴圈，當迴圈終止後，$i 的值為 11，而不是 10。這是因為 $i++ 的敘述是在迴圈裡的程式片斷執行完成後才被執行，然後再回到迴圈的開頭做 $i <= 10 的邏輯判斷，因為先遞增 1 後再進行條件判斷，所以 $i 最後跳出迴圈時的值為 11，而不是 10。

break 指令的用法

break 指定用來跳出目前的迴圈，通常是用在利用 if 敘述判斷到某個符合的條件，而必須馬上跳出迴圈時才使用。要注意的是，break 只能跳出一層迴圈，而且 break 只能用在[[迴圈]]敘述裡。

範例：

```
for ($i = 0; ;$i++) {
   if ($i > 10) break;
}
```

在這個範例裡，for 敘述沒有中止條件。中止條件寫在迴圈的程式裡，利用 if 來判斷，當 $i > 10 時，則執行 break 跳出迴圈。

continue 指令的用法

continue 與 break 是相對的指令。break 中斷目前執行的迴圈，continue 則是回到迴圈的開頭，重新執行迴圈。

例如：

```
for ($i = 0; $i < 10; $i++) {
   if ($i == 5) continue;
   printf("%d", $i);
}
```

輸出結果為：

```
12346789
```

當 if 判斷到 $i 等於 5 時，就跳回迴圈的開頭執行。

switch 敘述

switch 是很好用的多條件判斷敘述，跟 if...elseif...elseif... 結構相同。
switch 語法：

```
switch (EXPRESSION) {
   /* 當 EXPRESSION 的值為 1 時，則執行這裡的敘述。*/
  case 1: statements; break;

   /* 當 EXPRESSION 的值為 2 時，則執行這裡的敘述。*/
   case 2:statements; break;

   /* 當 EXPRESSION 的值都未出現在以上的 case 時，則執行這裡的敘述。*/
   default: statements; break;
}
```

範例：

```
switch ($i) {
   case 'A': $grade = 90;
```

```
                break;
   case 'b':
   case 'B': $grade = 80;
               break;
   default: $grade = 60;
              break;
}
```

- PHP 常數與變數

數字的寫法
數字分為十進位、八進位與十六進位三種寫法：

(1) 1234 - 一般我們習慣的十進位
(2) 01234 - 開頭為 0 (zero) 表示這是一個八進位數字
(3) 0x1234 - 開頭為 0x (zero eks) 表示這是一個十六進位數字

另外，double (浮點數) 的寫法也是和平常的習慣一樣，例如：

-12.3 3.5 7.0001 0.0000054 10000.1

這五個數字都是浮點數正確的寫法。

PHP 也支援科學記號寫法，可用來表示較大或較小的數值，例如：

12.5E-5 (12.5x10-5)
1.3E+3 (1.3x10+3)

分別為 .000125 (0.000125) 與 1300。

字串的寫法

字串以單、雙或倒引號圍住分別有不同的義意。

單引號

例如：

```
$str = 'An apple a day keeps the docter away.'
```

當字串出現 ' 符號時，必須加上 \ 斜線：

```
'I'm Jollen'
```

應改成：

```
'I\'m Jollen'
```

才對，其中 \' 即稱為跳脫字元 (escape character)。

雙引號

以雙引號圍住的字串 PHP 會對該字串做 variable interpolation 的動作，亦即做變數的取代：

```
$name = "Jollen";
echo 'Name: $name';
echo "Name: $name";
```

執行結果為：

```
Name: $name
Name: Jollen
```

在雙引號裡的字串如果有 $ (dollar sign)，只要改成跳脫字元的寫法即可：

```
$total = 12000
echo "Total: \$ $total";    //輸出 Total: $ 12000
```

在做 variable interpolation 時, 變數名稱是以一個以上空格做為界線, 例如：

```
$n_file = 5;

if ($n_file == 1) {
    echo "There are $n_file.";
} else {
```

```
    echo "There are $n_files.";
}
```

當 $n_file 不為 1 時，"There are $n_files." PHP 所看到的變數為 $n_files，而不是正確的 $n_file，所以必須改成：

```
$n_file = 5;

if ($n_file == 1) {
    echo "There are $n_file.";
} else {
    echo "There are {$n_file}s.";
}
```

單引號內的雙引號，或是雙引號內的單引號都視為有效字元，不需使用跳脫字元，例如：

```
echo "I'm a happy bird.";
echo 'I\'m a happy "bird"!';
```

輸出結果為：

```
I'm a happy bird.
I'm a happy "bird"!
```

反引號

利用反引號可以執行 UNIX 下的命令，並傳回執行結果。例如：

```
echo `ls -l *.txt`;
```

表示將 ls -l *.txt 命令的執行結果輸出，以反引號圍住的字串為要執行的 UNIX 指令。

如何使用 PHP 的資料型態

PHP 的變數屬鬆散資料型別，變數的型態是在計算時動態 (dynamic) 決定的 PHP 的鬆散資料型別，即我們給定什麼值，該變數即為什麼型別，或是如何使用

該變數，該變數即為適當的型別，例如：

```
$foo = "0";  $foo 為 string (ASCII 48)
$foo++;  $foo 變成 string "1" (ASCII 49)
$foo += 1;  $foo 變成 integer (2)
$foo = $foo + 1.3;  $foo 變成 double (3.3)
$foo = 5 + "10 Little Piggies";  $foo 為 integer (15)
$foo = 5 + "10 Small Pigs";  $foo 為 integer (15)
```

字串型態轉數值型態

當我們給變數的值是利用雙引號括住數值或字串時，就是指定一個字串給變數，例如：

```
$a = "Hello!";
```

$a 變數為字串型態。請看底下的範例：

```
<?php
$a = "hello!";
echo $a;
?>
```

輸出結果為：

```
hello!
```

PHP 的變數是在執行時才決定型態，因為字串也可以用來做計算，很奇怪吧！PHP 將字串拿來做運算時，會依據底下二個原則設法將字串轉成可以計算的型態：

字串中包括 "."、"e" 或 "E" 時轉換成 double 型別，否則轉換為 integer
無法轉換時則為 0
範例：

```
$foo = 1 + "10.5"; $foo 為 double (11.5)
$foo = 1 + "-1.3e3"; $foo 為 double (-1299)
$foo = 1 + "bob-1.3e3"; $foo 為 integer (1)
$foo = 1 + "bob3"; $foo 為 integer (1)
```

$foo = 1 + "10 Small Pigs"; $foo 為 integer (11)
$foo = 1 + "10 Little Piggies"; $foo 為 integer (11)
$foo = "10.0 pigs " + 1; $foo 為 int (11)
$foo = "10.0 pigs " + 1.0; $foo 為 double (11)

Type Juggling (型態轉換競爭)

變數在做運算時，例如使用 "+"，當 expression 包含各種不同的型態時，就會有 Type Juggling 的動作發生，例如：

$foo = "0"; $foo 為 string "0" (ASCII 48)
$foo++; $foo 為 string "1" (ASCII 49)
$foo += 1; $foo 變成 integer (2)
$foo = $foo + 1.1; $foo 變成 double (3.1)
$foo = 5 + "15 Persons"; $foo 的運算結果為 integer (20)
$foo = 5 + "10 Big Pigs"; $foo 的運算結果為 integer (15)

又如：

```
$a = 5;                 // $a 的型別為 integer
$a[0] = "Hi!";    // $a 的型別變成 array
```

這種型別的改變即稱為 "Type Juggling"。

區域變數

在 function 裡初始化的變數即區域變數。為什麼叫區域變數呢？因為區域變數只有在 function 裡可以被「看見」，請看底下的範例：

```
function sum() {
   $a = 1;
   $b = 2;
   echo $a+$b;
}
echo "<br>". $a;
sum();
```

執行結果：

```
3
0
```

第二個輸出的結果為 0, 這是因為 function 裡的 $a 只有在 function 裡才能被看到，在 function 外區域變數就不能被看見。

全域變數

在區域變數範圍之外所宣告的變數即全域變數，例如：

```
$a = 1;

function sum() {
echo $a;
}

sum();
```

這段程式碼執行後不會有任何輸出，可是 $a 不是一個全域變數嗎？請記得一點，因為 PHP 的變數是不須經過宣告的，所以 function 裡的 $a 其實仍然是一個區域變數。

區域變數的可見度會蓋掉全域變數，所以 sum() 所 echo 出的 $a 變數是一個區域變數, 那該如何告訴 function 變數是一個全域變數呢？利用 global 關鍵字即可：

```
$a = 1;

function sum() {
   global $a;

   $a = $a*100;
}

sum();
echo $a;
```

執行結果：

100

第一段程式碼其實存在了二個變數，一個是全域變數 $a，另一個則是區域變數 $a。在第二段程式碼裡，則只有一個全域變數 $a。

對於全域變數另外一個重點就是，倒底全域變數的範圍為何？在 PHP 裡，全域變數也稱為 page-scoped 變數, 亦即在同一個檔案裡的 PHP 程式都能看到這個全域變數。

靜態變數

區域變數生命期是在函數執行期間，隨函數的執行結束而結束，而靜態變數的生命期是隨整個 PHP 程式結束而結束，但可見度只有該函數。利用關鍵字 static 來宣告靜態變數：

```
function sum() {
   static $a = 1;

   if ($a < 10) {
   echo $a;
   $a++;
   sum();
}

sum();
```

輸出結果：

123456789

區域或全域變數都不是靜態變數，因為函數執行結束後，變數的值並不會被保留。而所謂的靜態變數意思就是說，當函數執行結束後，該變數的值仍然會被保留，因此第二次函數該函數時，靜態變數之前的值仍然存在。

常數的定義

使用者自定常數可使用 define() 函數，這些常數定義後，包括 PHP 裡事先定義好的常數，都不能再被重新定義。例如我們要定義 PI 常數的值為 3.14159：

```
define("PI", 3.14159);
echo PI;              // 輸出為 3.14159
```

這裡的 "PI" 被我們定義成常數，因此以後只要提到 PI，指的就是 3.14159，例如：

```
echo PI*10;
```

輸出結果等於 31.4159 (3.14159*10)。

- PHP 函數設計

如何定義 function
PHP 的函數定義是利用 function 關鍵字，語法：

```
function func_name(傳入參數 1, 傳入參數 2, ...) {
...
    程式區
...
}
```

函數可以讓我們的程式更具可讀性，也符合模組化程式設計的原則。底下是一個定義函數的範例：

```
function IsAlpha($x) {
if ($x >= 'a' && $x <= 'z') return TRUE;
if ($x >= 'A' && $x <= 'Z') return TRUE;
return FALSE;
}
```

注意事項：

在 PHP3 裡，函數必須在第一次被呼叫之前就被定義，否則會出錯，在檔案裡，則是循序順序，也就是，假如　IsAlpha() 要被呼叫，則在呼叫他的那行程式碼之「前」，必須有定義 IsAlpha() 函數的程式碼。不過，PHP4 裡則沒有這個

限制。

只要是符合 PHP 語法的敘述，都可以寫在 function 裡，包括其它的 function 或 class。另外，由於 PHP 並不支援 function overloading，所以我們無法重新宣告一個已經宣告過的函數。

函數的命名規定

使用者自定函數的名稱命名有三個限制：

不能與 PHP 的函數名稱同名。例如名稱不能為 mysql_pconnect()。
函數名稱不能以數字開頭。
函數名稱不能使用 "." (period)，例如 add.integer() 就是一個錯誤的函數名稱。
參數傳遞與傳回值

PHP 支援 call by value 與 call by reference 兩種傳遞參數的方法，並且在函數執行結束後，可以利用 return 傳回一個值給父程式。

call by value

call by value 是父程式直接將值或變數傳給函數，因此該數值或變數被儲存於兩個不同的記憶體位置。當傳入一個變數時，在函數裡改變傳入的參數值對父程式的變數並不會造成影響。

範例：

```
function sum($x, $y) {
   $x += $y;
   return $x;
}

$x = 25;
$y = 10;
echo sum($x, $y);         // 輸出 35;
echo $x;                        // 輸出 25
```

因為參數的傳遞使用 call by value 的方式，因此父程式的 $x 與 sum() 函數

的 $x 變數不會互相影響。這是因為這二個變數是分別位於兩個不同的記憶體位置的關係。

call by reference

call by reference 是一種非常好用的機制，以 C++ 的程式碼為例，底下是一個將大寫字母轉小寫的範例：

```
char &UpperAlpha(char &x) {
   return
   (x >= 'A' && x <= 'Z') ? x : x|=0x20;
}

void main(void)
{
   char x = 'Q';
   clrscr();
   printf("\nx = %c", x);
   printf("\nReturn: %c", UpperAlpha(x));
   printf("\nx = %c", x);
}
```

x 為 'Q' 時，輸出為：

```
x = Q
Return: q
x = q
```

當 x 為 '9' 時，輸出為：

```
x = 9
Return: 9
x = 9
```

利用 call by reference 傳入一個變數參考時，等於是將指向該變數的指標傳給函數，在函數裡改變該參考等於改變原來的變數值，因此可以達到改變原來變數值的效果。利用 call by reference 傳遞傳數時，傳入的變數與函數裡的參數變數位於相同的記憶體位置。

在 PHP 程式裡如果要傳遞參考 (call by reference) 的話，有二種做法，我們可選擇適合我們使用的方法：

1. 呼叫函數時在變數前加上 &，例如：

```
add(&$x, $y);
```

此時 add() 的寫法沒有什麼不同：

```
function add($x, $y) {
   $x += $y;
}
```

2. 在函數的參數加上 &，例如：

```
function add(&$x, $y) {
$x += $y;
}
```

此時呼叫函數的寫法：

```
add($x, $y);
```

二種寫法實際上是有所不同的。第一種寫法是呼叫函數時在變數前加上 &，種寫法可以讓我們在呼叫函數時才決定是否要採用 call by reference 的方式傳遞參數。而第二種寫法是在函數的傳入值前加上 &，這種寫法會強迫使用 call by reference 的方式傳遞參數，而不管呼叫函數時是採取 call by value 或是 call by reference 的方式。

善用二種方式間的差異可以幫助我們維護程式的品質。例如，當我們的函數必須替父程式維護傳入變數值時，這時就可以強迫函數的傳入值採取 call by reference 的方式，如此就算我們呼叫函數時手誤少打了一個 "&"，也可以正確採取 call by reference 的方式。

參數預設值

參數預設值就是，當我們沒有傳入值給該參數時，該參數即以預設值來工作，例

如：

```
function add($x = 1, $y = 2) {
return $x+$y;
}

echo add();         // 結果為 3
```

上面這個範例中，因為我們並沒有傳入值給參數，所以 $x 與 $y 的預設值即為 1 與 2，所以傳回的結果為 3。

上預設值的參數必須全部靠右，例如：

```
function add($x, $y = 1, $z = 2) {
   return $x+$y+$z;     //傳回值 (return value)
}

echo add(5);                  // 結果為 8
```

傳回值

在前面的範例中，我們直接以 return [value] 的方式將值傳回給父程式。由於一個函數只會有一個有效的 return，所以只能有一個傳回值。

如何傳回多個值

如果要傳回多個值，可以利用傳回陣列的小技巧來達成，例如：

```
function test() {
   return array(10, 20, 30, 40, 50);
}
```

因為傳回給父程式的是一個陣列，因此父程式必須利用 list() 函數才能將陣列的值分別指定給變數。

範例：

```
list($a, $b, $c, $d, $e) = test();
```

```
echo $a;         // 輸出為 10
echo $b;          // 輸出為 20
echo $c;          // 輸出為 30
echo $d;          // 輸出為 40
echo $e;          // 輸出為 50
```

可變函數名稱

一般函數名稱在定義函數時即決定好，但 PHP 提供一種技術，可以讓我們將函數的名稱存放在一個變數裡，當呼叫這個變數時，即等於呼叫變數值相對應的函數。這種可以利用變數指定不同函數名稱的技術即稱為可變函數名稱。

底下是一個簡單的可變函數應用範例：

```
function male() {
   ...
}
function female() {
   ...
}
if ($sex == "male")
   person$ = 'male';
else
   person$ = 'female';

$person();
```

我們可在程式裡變態改動可變函數的名稱，如此即可完成呼叫同一個函數，但執行不同函數的效果。

- **PHP 陣列**

如何使用陣列
在 PHP 裡要使用陣列時，只要將變數進行「初始化」陣列時的程序即可。

範例：

```
$names[0] = "Jollen"
```

```
$names[1] = "Jordan"
$names[2] = "Kitty"
$names["howmany"] = 3;
```

scalar array 與 associative array

在上面的例子中，我們看到：

```
$names[3] = "Kitty"
```

是以 3 這個數值做索引，以數值做索引 (index, key) 的陣列稱為 scalar array。

另外一個例子：

```
$names["howmany"] = 3;
```

以字串做索引的陣列我們稱為 associative array。

多維陣列

多維陣列的初始化與一維陣列方法相同，例如：

```
$alpha[0][0] = "A";
$alpha[0][1] = "B";
$alpha[1][0] = "C";
$alpha[1][1] = "D";
```

這種寫法等於：

```
$alpha[0][] = "A";
$alpha[0][] = "B";
$alpha[1][] = "C";
$alpha[1][] = "D";
```

陣列的內部指標

PHP 裡的陣列事實上是利用資料結構中的雙向鏈結串列來維護的，因此我們可以

利用 next() 與 pre() 函數將陣列的內部指標往前或往後一個元素。陣列裡的指標除了利用 PHP 提供的函數外，外界無法直接去改變陣列的內部指標。

PHP 陣列函數 - array()、count()、current()、list()、next()、pre()、reset()、key()

array()

用途：

建立一個陣列，依給定的參數（語法）傳回陣列。

範例：

```
$name = array("Jollen", "Paul", "Ketty");
```

相當於：

```
$name[0] = "Jollen";
$name[1] = "Paul";
$name[2] = "Ketty";
```

array() 亦可為用在巢狀式陣列：

```
$fruits = array(
   "fruits" => array("a"=>"orange","b"=>"banana","c"=>"apple"),
   "numbers" => array(1, 2, 3, 4, 5, 6),
   "holes" => array("first", 5 => "second", "third")
);
```

上面為建立一個二維陣列的範例。此例相當於：

```
$fruits["fruits"]["a"] = "orange";
$fruits["fruits"]["b"] = "banana";
$fruits["fruits"]["c"] = "apple";

$fruits["numbers"][0] = 1;
$fruits["numbers"][1] = 2;
```

```
$fruits["numbers"][2] = 3;
$fruits["numbers"][3] = 4;
$fruits["numbers"][4] = 5;
$fruits["numbers"][5] = 6;

$fruits["holes"][0] = "first";
$fruits["holes"][5] = "second";
$fruits["holes"][6] = "third";
```

array() 事實上並不是函數，而是 PHP 提供的一個語法。

count()

定義：

```
int count(mixed var);
```

用途：

傳回 var（通常為陣列）的元素個數，非陣列的變數則只有一個元素。沒果沒有這個 var 變數，則傳回 0；如果 var 不是陣列，則傳回 1。

範例：

```
$names = array("jollen", "nick", "frank");

echo count($names);
```

輸出：

```
3
```

current()

定義：

```
mixed current(array array);
```

用途：

傳回目前陣列裡的指標所指元素的「值」。每一個陣列都有一個內部的指標，指向其中的一個元素。陣列的所有元素利用雙向串列連接，這個指標便指向目前的元素。一開始這個指標是指到陣列的第一個元素，利用其它函數存取陣列時，便會改變這個指標，current() 便是傳回目前所指的陣列元素的值，但不會改變這個指標的位置。

傳回指標指向陣列的範圍之外的位置，便傳回 false。有一種情況要特別小心，當元素的值為 0 或是空字串 "" 時，也會傳回 false，此時可改用 echo() 函數。

list()

list() 並不是一個 function，而是像 array() 一樣，屬於 PHP 的語法。list() 用來一次給定多個值給多個變數。

範例：

```
<table>
<tr>
<th>Employee name</th>
<th>Salary</th>
</tr>
<?php

$result = mysql($conn, "SELECT id, name, salary FROM employees");
while (list($id, $name, $salary) = mysql_fetch_row($result)) {
   print(" <tr>\n".
   " <td><a href=\"info.php3?id=$id\">$name</a></td>\n".
   " <td>$salary</td>\n".
   " </tr>\n");
}

?></table>
```

next()

定義：

```
mixed next(array array);
```

用途：

傳回下一個陣列指標所指的值，注意是先把指標往下移，再傳回值。如果元素的值是 0 或 "" (空字串)，則傳回 false。當指標已指向陣列尾段，無法再往下移時，則傳回 false。

可利用 echo() 函數來觀察陣列中是否有 0 或 ""。

prev()

定義：

```
mixed prev(array array);
```

用途：

將陣列內部指標往前移前一位後，再傳回元素的值。

reset()

定義：

```
mixed reset(array array);
```

用途：將陣列的指標初始化，即移到第一個元素的位置，並且傳回第一個元素的值。

key()

定義：

```
mixed key(array array);
```

用途：

傳回目前 associative array 中的 key，即陣列內部指標所指元素位置的 key。

配合 reste()、next() 的綜合範例如下：

```
$fruits =
array("d"=>"lemon","a"=>"orange","b"=>"banana","c"=>"apple");
arsort($fruits);
for(reset($fruits); $key = key($fruits); next($fruits)) {
   echo "fruits[$key] = ".$fruits[$key]."<br>";
}
```

輸出：

```
fruits[a] = orange
fruits[d] = lemon
fruits[b] = banana
fruits[c] = apple
```

迴圈由第一個元素開始拜訪所有的元素，到最後一個元素結束。

- PHP 良好習慣

在這裡為止我們已經學會所有 PHP 的基本觀念了，接下來我們就要帶著我們的武器，到戰場上實際操練一番。在這之前，要說明的是：養成良好的程式寫作習慣是非常重要的事，關於 PHP，底下是幾個不好的程式寫作習慣與建議：
變數值的輸出

不好的寫法：echo "$var";
建議改寫成：echo $var;
print 的寫法

不好的寫法：print("Hello!");
建議改寫成：print "Hello!";
HTML 標籤的輸出

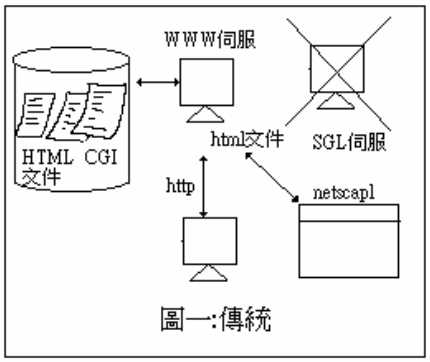
不好的寫法：echo "PHP";
建議改寫成： ... ?>PHP<? ...

陣列的初始化寫法

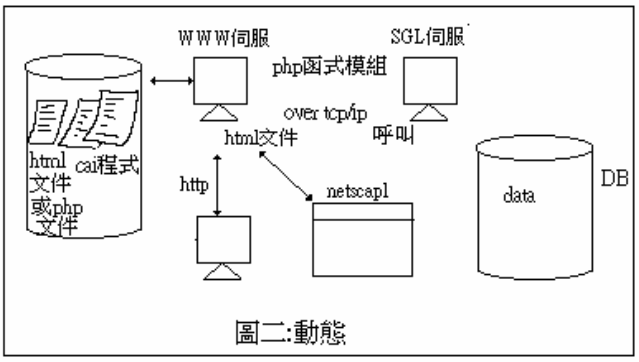
不好的寫法： $a[0]=1; $a[1]=2; $a[2]=3;
建議改寫成：$a = array(1,2,3);
條件的判斷

不好的寫法：if($a>1) { $b='a'; } else { $b='b'; }
建議改寫成：$b = ($a>1) ? 'a':'b';
對函數傳值做錯誤檢查

未做錯誤檢查的寫法：$result=mysql_query(...);
建議改寫成：$result=mysql_query(...) OR die(...);



圖一:傳統



圖二:動態

參考文獻

參考資源

- W3C 網站：www.w3c.org
- Apache Project 網站：www.apache.org
- PHP 網站：www.php.net
- Javascript 網站：
  - developer.netscape.com/viewsource/goodman_cssp/goodman_cssp.html
  - www.resource-centre.net/javascript/
- MySQL 網站：www.mysql.com

參考資料

- 「專業 PHP 程式設計」，許鳴程 譯，基?
- 「PHP4+MySQL 完整自學方案」，趙啟志 著，博碩文化
- 「JavaScript 應用程式設計」，陳建勳 譯，美商歐萊禮