# 1.7 JavaScript 撰寫慣例

HTML、CSS 和 JavaScript 三者程式碼可以放在同一個 HTML 文件檔內，也可以個別獨立存放。若 CSS 或 JavaScript 的程式碼敘述很少，可寫在同一個 HTML 文件內；若 CSS 或 JavaScript 的程式碼敘述很多，建議各自撰寫存放，如此程式的維護比較容易，也可以供多個檔案重複使用。

## 1.7.1 將 HTML、CSS 和 JavaScript 寫成一個文件

**範例：**

將 HTML、CSS 和 JavaScript 直接嵌入到一個 HTML 檔中。當用滑鼠按一下網頁上的文字時，透過 JavaScript 來改變文字的字體大小。

**程式碼** FileName : action.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>互動範例</title>
    <style>
      .box {
        font-size: 16px;
      }
    </style>
  </head>
  <body>
    <p class="box" id="myBox" onclick="change()">點按此使文字放大</p>
    <script>
      function change() {
        var big = document.getElementById('myBox');
        big.style.fontSize = 40 + 'px';
      }
    </script>
  </body>
</html>
```

## 執行結果


## 說明

1. 本程式的內容對初學者而言，可能難度較高。對程式內容及說明不瞭解沒關係，在往後的章節會陸續介紹。本範例用意在呈現 HTML、CSS 和 JavaScript 三者程式碼合體互動的情形。

2. 第 5~9 行：在 HTML 文件的 <head> 標籤中，將 CSS 的樣式設定在 style 標籤裡。其功能是將文字字體大小設為 16 px。

3. 第 12 行：<p> 元素是成對的容器標籤，可包含一個段落文字顯示在網頁，該文字內容為「點按此使文字放大」。<p> 元素有三個屬性設定：
   ① class 屬性的屬性值為 "box"。是要套用 CSS 的「.box」樣式 (第 6~8 行)，即使得 <p> 標籤元素顯示的文字字體大小為 16 px。
   ② id 屬性的屬性值為 "mybox"。HTML 文件每一個元素，皆是一個物件，而 id 是該物件的身分代號，id 屬性值具有唯一性。
   ③ onclick 是事件屬性，觸動該事件就執行由 JavaScript 程式所撰寫的 change() 函式 (第 14~17 行)，觸動的方式是按一下滑鼠左鍵。

4. 第 15~16 行：在 HTML 文件的 <script> 標籤中編寫 JavaScript 程式，這裡寫了一個 change() 函式。該函式被呼叫時，會執行下列動作：
   ① 第 15 行先宣告一個變數 big，再將 id 為 'mybox' 的物件 (<p> 元素) 指派給 big 變數。
   ② 第 16 行將 big 變數內的文字字體設為 40 px。使得 <p> 元素的文字字體大小由 16 px 變成 40 px，故網頁上顯示的文字字體變大了。

## 1.7.2 將 CSS 和 JavaScript 寫成外部檔案

前面的 HTML 文件檔「action.html」，使用 <style> 元素內嵌 CSS 程式碼，使用 <Script> 元素內嵌 JavaScript 程式碼。而這些內嵌的 CSS 和 JavaScript 程式碼可以另外寫成外部檔案。若將文件檔各自存放，則 HTML 檔案副檔名為 .html、CSS 檔案副檔名為 .css、JavaScript 檔案副檔名為 .js。

**範例**：

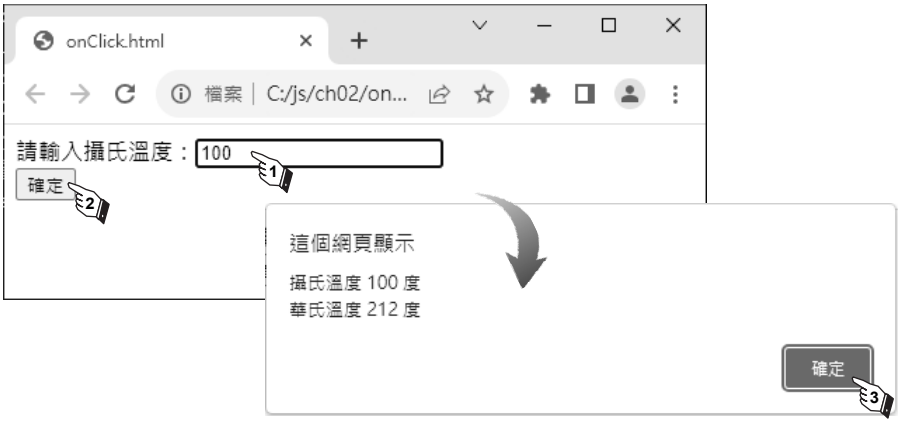
用 VS Code 編輯區器將前面範例「action.html」中的 CSS 和 JavaScript 程式碼，分別寫成外部檔案，三個檔案分別取名為「action-2.html」、「action-2.css」、「action-2.js」。

**程式碼** FileName : action-2.html

```
01 <!DOCTYPE html>
02 <html>
03   <head>
04     <title>互動範例</title>
05     <link rel="stylesheet" href="action-2.css">
06   </head>
07   <body>
08     <p class="box" id="myBox" onclick="change()">點按此使文字放大</p>
09     <script src="action-2.js"></script>
10   </body>
11 </html>
```

# 2.9 JavaScript 與表單互動

以下是一個最簡單的範例，實作如何使用 JavaScript 與 HTML 表單進行基本的互動。在這個範例建立包含一個輸入框和一個按鈕的表單。當用戶在輸入框輸入資料並點按按鈕時，瀏覽器將表單資料傳送給 JavaScript 處理，最後在網頁上彈出訊息框來顯示處理後資訊。




程式設計說明如下：

1. 表單元素內容

```
01 <form>
02      請輸入攝氏溫度：<input type="text" id="textC"><br>
        <input type="button" value="確定" onclick="showTemp()">
03 </form>
```

① 第 1 行：<form> 元素內沒有設定 action 和 method 屬性，表單資料預設以 GET 的方式傳送給目前的瀏覽器網頁處理。

② 第 2 行：這是一個段落內容，開頭有文本「請輸入攝氏溫度：」，接著一個 text 元件(輸入框)。遇到 <br> 換行。段落第二行再顯示 button 元件 ( 確定 ) 按鈕。

③ 點按「確定」按鈕會觸發 JavaScript 程式所編寫的函式，所以在 button 元件內要設定 onClick 屬性，並指定觸發 showTemp() 函式。

④ text 元件 (輸入框) 內所輸入的資料，在 JavaScript 程式中會參用到，所以要設定 id 屬性，做為宣告變數時識別用途。

2. 在 JavaScript 程式中如何參用 HTML 元件的 id。

```
var txtC = document.getElementById('textC');
```

① 宣告一個變數 txtC，其變數值來自 document.getElementById('textC') 指定。其中 'textC' 是 HTML 元件的 id 值，所以變數 txtC 的內容是 'textC' 元件，變數 txtC 的資料型別為 object (物件)。

② 變數名稱 txtC 和 text 元件的 id 值 'textC'，為了避免命名的困擾，兩者可以取一樣的名字是不會有衝突。

3. 在 JavaScript 程式中如何取得 text 元件 (輸入框) 的填寫資料。

```
var numC = Number(txtC.value);
```

① 變數 txtC 是一個物件，該物件的文字內容存放在 value 屬性中。用「物件.屬性」語法取得屬性值，故用「txtC.value」存取文字內容。

② 因「txtC.value」敘述取得的資料屬 String 型別，故用 Number() 函式將其轉型為數值資料，再指定給 numC 變數。故 numC 為 Number 型別變數，在此代表攝氏的溫度值。

4. 將攝氏溫度值換算為華氏溫度值，再指定給變數 numF。

```
var numF = numC * 9/5 + 32;
```

5. 使用合併指定運算子來組合資訊字串，然後用網頁彈出的訊息框來顯示處理後的訊息。

```
var msg = '攝氏溫度 ' + numC + ' 度 \n';
msg += '華氏溫度 ' + numF + ' 度';
window.alert(msg);
```

完整程式碼如下：

**程式碼** FileName : onClick.html

```
<!DOCTYPE html>
<html>
  <body>
    <form>
      請輸入攝氏溫度：<input type="text" id="textC"><br>
      <input type="button" value="確定" onclick="showTemp()">
    </form>
    <script>
      function showTemp() {
        var txtC = document.getElementById('textC');
        var numC = Number(txtC.value);
        var numF = numC * 9/5 + 32;
        var msg = '攝氏溫度 ' + numC + ' 度 \n';
        msg += '華氏溫度 ' + numF + ' 度';
        window.alert(msg);
      }
    </script>
  </body>
</html>
```

如果回傳給瀏覽器資料要顯示在網頁上，要如何處理。


程式設計說明如下：

1. 使用 <p> 段落元素來顯示文字

```
  <p id="output"></p>
```

JavaScript 程式中會設計使用 <p> 元素來顯示文字，所以要設定 id 屬性，做為宣告變數時識別。

2. 在 JavaScript 程式中如何在 <p> 段落元素寫入文字。

```
var txt = document.getElementById('output');
txt.textContent = '按鈕 被點按了';
```

① 宣告一個變數 txt 代表 id 值為 'output' 的物件。

② HTML 元素的文本內容由 textContent 屬性來存放。

完整程式碼如下：

**程式碼** FileName : textContent.html

```
01 <!DOCTYPE html>
02 <html>
03   <body>
04     <input type="button" value="按鈕" onclick="showText()">
05     <p id="output"></p>
06     <script>
07       function showText() {
08         var txt = document.getElementById('output');
09         txt.textContent = '按鈕 被點按了';
10       }
11     </script>
12   </body>
13 </html>
```

# 3.2 if 選擇結構

if 選擇結構可分為單向及雙向選擇結構，再由這兩種基本結構衍生出多向和巢狀等其他型式。撰寫程式時，可以依照不同的需求，挑選合適的選擇敘述。

## 3.2.1 條件運算式

一個選擇結構至少要包含一則條件運算式，條件運算式亦可簡稱為「條件式」，條件式可以使用關係運算式、一般算術運算式或邏輯運算式，選擇結構遇到條件式時，會進行運算並指出該條件式結果為真 (true)，或是為假 (false)。

在條件運算式中，如何判斷運算的結果是真或是假？JavaScript 所根據的原則如下：

- 運算結果是一個數值時，若此數值等於 0，則是假，否則就是真。
- 運算結果是一個字串時，若此字串等於空字串 ''，則是假，否則就是真。
- 運算結果是一個布林值時，若是 false 就是假；反之，若是 true 則為真。

**簡例** 條件運算式簡例。(test01.html)

```
<script>
  if(8 + 9)                        // 數值
    window.alert('true');          // 輸出 true
  else
    window.alert('false');

  if('')                           // 空字串
    window.alert('true');
  else
    window.alert('false');         // 輸出 false

```

```
12   window.alert(0 == (3 % 2));  // 輸出 false
13 </script>
```

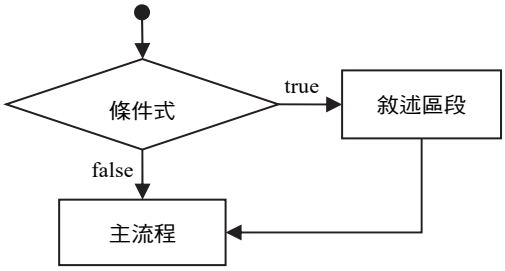
基本型的條件運算式是由比較運算子和兩個運算元所組成的；但如果條件比較複雜，需針對多個條件運算式一起做判斷時，此時要利用「邏輯運算子」來作連結，組合成進階型的條件式。其運算後的結果只會有兩種，不是 false， 就是 true。

**簡例** 進階型條件運算式簡例。(test02.html)

```
01 <script>
02   var x = 18;
03   if((x > 12) && (x < 20))    // 判斷 x 是否為 13~19 之間的任一數
04     window.alert('true');     // 輸出 true
05   else
06     window.alert('false');
07
08   if((x <= 12) || (x > 65))   // 判斷 x 是否為小於 12 或大於 65 的任一數
09     window.alert('true');
10   else
11     window.alert('false');    // 輸出 false
12 </script>
```

## 3.2.2 單向選擇結構

「單向選擇結構」敘述是當條件運算式運算結果為 true (真) 時，執行條件運算式後面的敘述區段；當運算結果為 false (假) 時，則結束選擇結構，繼續執行主流程之敘述。程式流程圖和語法如下：




**語法**

```
語法 1：if (條件運算式)  window.alert ("true") ;    // 單行敘述
語法 2：if (條件運算式)
            window.alert ("true") ;    // 單行敘述
語法 3：if (條件運算式)
        {
            ..... ;                          // 敘述區段
            ..... ;
        }
```

## 說明

1. 條件運算式必須用小括號「( )」括起來。
2. 若敘述區段內有多行敘述時，必須用大括號「{ }」括起來。若僅有一行敘述，則可省略大括號。為了提高程式碼可讀性，敘述區段宜向後縮排一個階層。
3. 在僅有單行敘述的情況下，條件運算式可以和單行敘述合併寫成一行，如語法 1。

**簡例** 單向選擇結構簡例。(if.html)

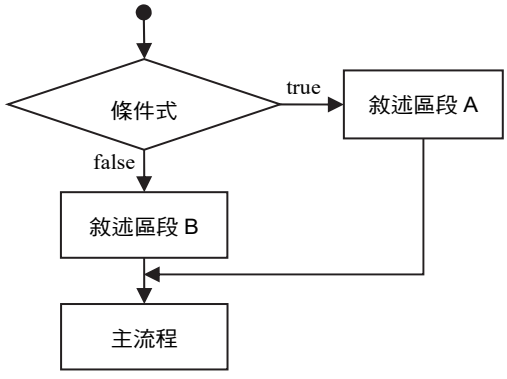
```
01 <script>
02   var x = -6;
03   if (x < 0)
04     x = -x;
05   window.alert(x);       // 輸出 6
06 </script>
```

**說明**

1. 條件運算式判斷變數 x 如果小於 0，則執行第 4 行，以一元運算子「-」取該變數的絕對值，之後再執行第 5 行；如果變數 x 如果大於 0，則直接跳到第 5 行，執行輸出敘述。

## 3.2.3 雙向選擇結構

「雙向選擇結構」敘述有如口語中的「如果…就…否則…」，當條件運算式運算結果為 true (真) 時，執行條件運算式後面的敘述區段 A；當運算結果為 false (假) 時，則執行 else 後面的敘述區段 B。程式流程圖和語法如下：




**語法**

```
if (條件運算式) {
   window.alert ("true") ;     // 條件式為真時的敘述區段
} else {
   window.alert ("false") ;    // 條件式為假時的敘述區段
}
```

**簡例** 雙向選擇結構簡例。(ifelse.html)

```
01 <script>
02   var x = 2;
```

```
03   if (x == 0)
04     window.alert('除數不得為 0');
05   else
06     window.alert(10 / x);
07 </script>
```

### 說明

1. 條件運算式判斷變數 x 如果等於 0，則執行第 4 行，執行 if 敘述區段，顯示「除數不得為 0」訊息框；如果變數 x 不等於 0，則直接跳到第 6 行，執行 else 敘述區段，顯示除法運算後的商。

## 3.2.4 條件運算子

條件運算子是 JavaScript 語法中唯一的「三元運算子」，所謂三元運算子是指該運算子執行時，需要有三個運算元才能正確執行。條件運算子是 JavaScript 中很特殊的運算子，可以依照運算式的結果，將不同的資料設定給指定的變數。換言之，條件運算子可以用來簡化 if … else … 選擇結構，而且該運算子也能支援巢狀運算。語法如下：

**語法** 語法： var 變數 = 條件運算式 ? 變數值 1 : 變數值 2;

**簡例** 條件運算子簡例。(test03.html)

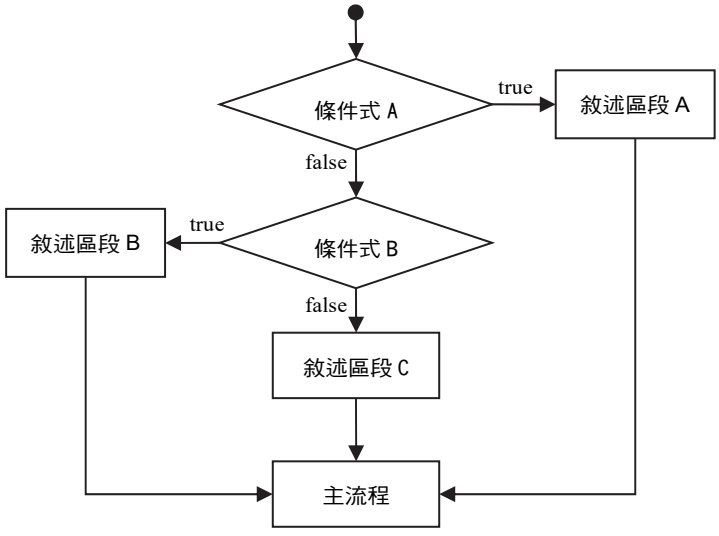
```
01 <script>
02   var sex = 'Male';
03   var title = (sex == 'Male') ? '先生' : '女士';
04   window.alert(title + '你好!');
05 </script>
```

### 說明

1. 條件式運算的結果為 true，就將問號後方的字串 '先生'，指定給等號左側的變數 title；如果是 false，則就將冒號後方的字串 '女士'，指定給等號左側的變數 title。

## 3.2.5 巢狀選擇結構

「巢狀選擇結構」是指在一個選擇結構的 if 敘述區段中或者是在 else 敘述區段中再置入另一個選擇結構。這種結構可以讓原本的一個或兩個選項擴展成多個選項，應用層面會更加廣大。巢狀選擇程式流程圖和語法如下：




**語法**

```
if (條件運算式 A) {
   window.alert ("敘述區段 A") ;     // 條件式為真時的敘述區段 A
} else {
   if (條件運算式 B)                  // 條件式為假時的敘述區段
      window.alert ("敘述區段 B") ;  // 巢狀選擇的分支敘述區段 B
   else
      window.alert ("敘述區段 C") ;  // 巢狀選擇的分支敘述區段 C
}
```

**範例：**

使用巢狀選擇結構來判斷使用者所輸入的 x、y 座標，位於哪一個象限。

**程式碼** FileName : quadrant.html

```
<!doctype html>
<html>
  <head>
    <title>QUADRANT</title>
  </head>
  <body>
    <script>
      var x = Number(window.prompt('請輸入 x 座標值：'));
      var y = Number(window.prompt('請輸入 y 座標值：'));
      if(x > 0) {
        if(y > 0)
          window.alert(x + "," + y + " 位於第一象限");
        else
          window.alert(x + "," + y + " 位於第四象限");
      }
      else {
        if(y > 0)
          window.alert(x + "," + y + " 位於第二象限");
        else
          window.alert(x + "," + y + " 位於第三象限");
      }
    </script>
  </body>
</html>
```

**執行結果**

# 12.2 儲存 Cookie 資料

## 12.2.1 Cookie 簡介

Cookie 是儲存在使用者瀏覽器的文字資料，通常是由伺服器透過 Set-Cookie 標頭 (header) 傳遞給瀏覽器。瀏覽器收到 Cookie 後會儲存起來，之後向伺服器請求 (request) 時會回傳 Cookie 作為識別。Cookie 可以用於身份認證、購物車、遊戲分數、使用者設定、廣告、追蹤使用者喜好…等方面，用來提供更好的網站訪問體驗或網站的統計資料。例如當使用者訪問網頁時，使用者的相關資訊和操作過程會記錄在 Cookie 中，在使用者下一次訪問時，網站可以由 Cookie 中取得之前的訪問記錄。

因為 Cookie 會附加在 HTTP 請求中，在瀏覽器和伺服器間相互傳送，所以會占用網路頻寬而影響效能。而且如果 Cookie 沒有加密保護，會有資料洩漏的危險。因為 Cookie 會嚴重影響效能，所以瀏覽器會對 Cookie 做些限制，例如 Cookie 的容量大小和個數。

## 12.2.2 Cookie 物件的常用屬性

在用戶端可以用 JavaScript，透過 document.cookie 物件來管理瀏覽器的 Cookie。document.cookie 物件中只能存放字串和數值等基本型別資料，陣列和物件等較特殊的資料無法儲存。

| 屬性 | 說明 |
|---|---|
| domain | 讀取或設定**網域名稱**，來指定哪些網域可存取此 Cookie。例如 domain = go.com 表該網域與子網域如 www.go.com、blog.go.com…等都能存取，若省略則預設為目前的網域。 |
| path | 讀取或設定**路徑**，用來指定哪個路徑與其子路徑可以存取這個 Cookie。例如設為 path=/ 表示整個網域都能存取，若省略時則預設為目前 URL 的路徑。 |

| 屬性 | 說明 |
|---|---|
| max-age | 讀取或設定 Cookie 的**有效秒數**。例如設為 86400 就是一天後失效，若為 0 或-1 會使 Cookie 立即失效。如果省略預設關閉瀏覽器時 Cookie 就失效。 |
| expires | 讀取或設定 Cookie 的**有效日期**，格式為 UTC 日期字串。如果省略預設關閉瀏覽器時 Cookie 就失效。 |
| Secure | 讀取或設定限定 Cookie 只能透過 https 通訊協定傳遞，省略時預設不區分 http 或是 https。 |
| HttpOnly | 讀取或設定禁止 JavaScript 存取 Cookie，來防止惡性的網路攻擊。 |

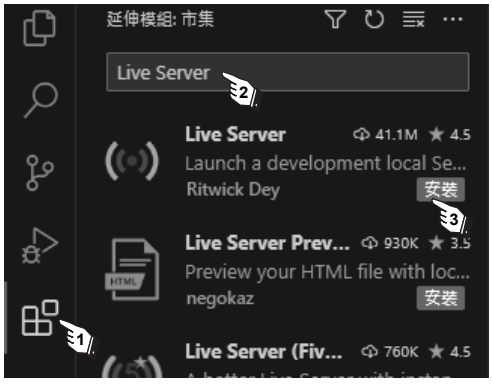
因為 Cookie 可能被攻擊，所以設定相關的屬性，來限制有效範圍、時間、協定做為保護。使用 Javascript 建立 Cookie 的語法如下：

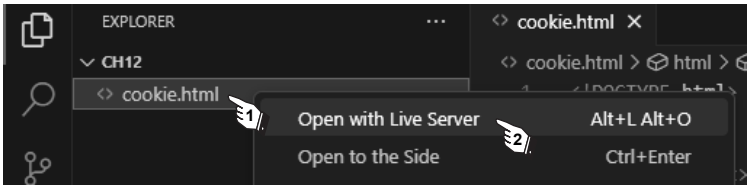
**語法**

```
document.cookie = 'Cookie 名稱 = Cookie 值 [; 屬性 = 屬性值; ...] ' ;
```

寫入 Cookie 時要使用字串，其中「Cookie 名稱 = Cookie 值」是必要參數，屬性設定則可以視需要而增加，參數間用分號「;」串接。如果寫入多個 Cookie 時只要名稱不同，舊的 Cookie 不會被覆蓋，新 Cookie 會串接到 document.cookie 中。

測試 Cookie 時，檔案必須由網頁伺服器執行，所以要安裝支援網頁伺服器的 Live Server 擴充套件。安裝步驟如右圖所示。執行時在 html 檔上按右鍵，執行「Open with Live Server」即可。






**簡例** 建立一個 Cookie，當瀏覽器關閉後會自動刪除。(cookie_1.html)

```
01 document.cookie = 'name=Jack';     // 未指定有效期限時預設為瀏覽器關閉後刪除
02 alert(document.cookie);            // 顯示：name=Jack
```


**簡例** 建立一個 Cookie，在網站的 cookie_1 資料夾下能存取，有效期間為兩天(2*24*60*60 = 172800)。

```
document.cookie = 'password=lucky168; path=/cookie_1; max-age=172800';
```

**簡例** 建立一個 Cookie 限定只能透過 https 傳遞，有效期間為一天。

```
01 let date = new Date();
02 date.setTime(date.getTime() + 24*60*60*1000);
03 document.cookie = 'id=love520;Secure;expires=' + date.toUTCString();
```

**Tips** 若同時設定 max-age 和 expires 屬性，會以 max-age 屬性優先。

**簡例** 設計一個函式，可以建立指定名稱和值的 Cookie。

```
01 function setCookie(name, value) {
02   document.cookie = name.toString()+ '=' + value.toString();
03 }
```