Appendix

Video/Audio API

C

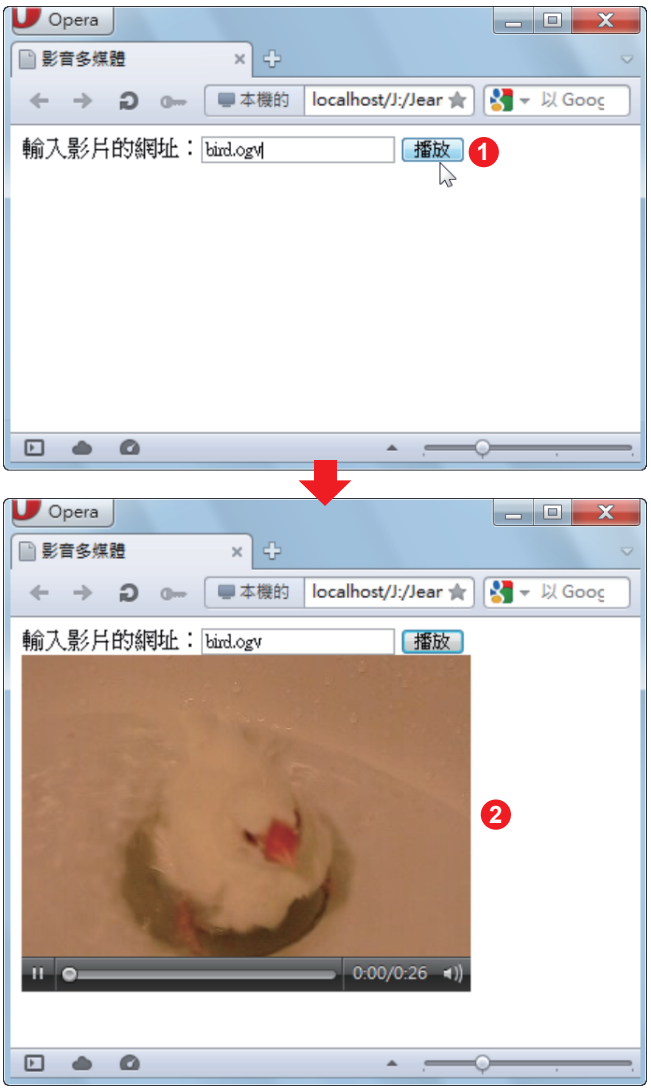
C-1 <video> 與 <audio> 元素的屬性與方法

<video> 與 <audio> 元素的 API 繼承自 HTMLMediaElement 媒體元素，而且這兩個元素的屬性、方法與事件幾乎都相同，除了 <audio> 元素沒有 width、height、poster 等屬性，下表為 <video> 與 <audio> 元素可以使用的屬性與方法 (參考自 HTML5 官方文件 http://www.w3.org/TR/html5/)。

屬性與方法	說明
error (唯讀屬性)	錯誤狀態，傳回值如下： • MEDIA_ERR_ABORTED (1)：因為使用者的操作而中止下載。 • MEDIA_ERR_NETWORK (2)：因為網路錯誤而中止下載。 • MEDIA_ERR_DECODE (3)：媒體資料解碼錯誤。 • MEDIA_ERR_SRC_NOT_SUPPORTED (4)：不支援的媒體資料格式。
src (屬性)	媒體資料的網址。
currentSrc (唯讀屬性)	實際播放之媒體資料的網址。
networkState (唯讀屬性)	網路狀態，傳回值如下： • NETWORK_EMPTY (0)：尚未初始化。 • NETWORK_IDLE (1)：尚未連結網路。 • NETWORK_LOADING (2)：下載中。 • NETWORK_NO_SOURCE (3)：不支援的格式，故不進行下載。
preload (屬性)	預先下載。
buffered (唯讀屬性)	傳回 TimeRanges 物件，表示媒體資料在緩衝區的範圍 (註：TimeRanges 物件有三個屬性 length、start、end，分別表示物件的長度、開始位置的秒數及結束位置的秒數)。
load() (方法)	下載媒體資源。
initialTime (唯讀屬性)	初始的播放秒數。
currentTime (屬性)	目前的播放秒數。
duration (唯讀屬性)	媒體資料的長度 (以秒數為單位)。

屬性與方法	說明
readyState (唯讀屬性)	就緒狀態，傳回值如下： • HAVE_NOTHING (0)：沒有資料。 • HAVE_METADATA (1)：有 metadata。 • HAVE_CURRENT_DATA (2)：有目前播放的資料。 • HAVE_FUTURE_DATA (3)：有即將播放的資料。 • HAVE_ENOUGH_DATA (4)：有足夠播放的資料。
paused (唯讀屬性)	是否暫停中，true 表示是，false 表示否。
defaultPlaybackRate (屬性)	預設的播放速度，初始值為 1.0，正數表示繼續往後播放，負數表示向前倒帶。
playbackRate (屬性)	播放速度。
played (唯讀屬性)	傳回 TimeRanges 物件，表示已經播放的時間範圍。
seekable (唯讀屬性)	傳回 TimeRanges 物件，表示可以往後定位的時間範圍。
ended (唯讀屬性)	是否播放結束，true 表示是，false 表示否。
autoplay (屬性)	是否自動播放，true 表示是，false 表示否。
loop (屬性)	是否反覆播放，true 表示是，false 表示否。
play() (方法)	開始播放。
pause() (方法)	暫停播放。
controls (屬性)	是否顯示控制面板，true 表示是，false 表示否。
volume (屬性)	音量，值為 0.0 (靜音) ~ 1.0 (最大聲)。
muted (屬性)	是否為靜音，true 表示是，false 表示否。
defaultMuted (屬性)	是否預設為靜音，true 表示是，false 表示否。
audioTracks (屬性)	傳回 AudioTrackList 物件，表示可用的音軌。
videoTracks (屬性)	傳回 VideoTrackList，表示可用的影片軌道。
canPlayType(*type*) (方法)	檢查是否能夠播放指定的媒體資料格式，傳回值如下： • ""(空字串)：參數 *type* 為 "application/octet-stream"，或瀏覽器不支援參數 *type* 指定的媒體資料格式。 • probably：瀏覽器支援參數 *type* 指定的媒體資料格式，且該格式有明確寫出 codecs 參數。 • maybe：瀏覽器支援參數 *type* 指定的媒體資料格式，但該格式沒有明確寫出 codecs 參數。

我們來簡單示範如何使用 JavaScript 存取 <video> 元素的屬性與方法，這個例子的瀏覽結果如下圖，使用者只要輸入影片的網址，然後按 [播放]，就會開始下載並播放影片，而且會顯示控制面板。在充分瞭解其中的技巧後，您就可以試著舉一反三，量身訂做自己專屬的媒體播放程式。



1 輸入影片的網址後按 [播放] 2 開始下載並播放影片

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>影音多媒體</title>
06:    <script>
07:      function playVideo() {
08:        var myVideoURL = document.getElementById("myVideoURL").value;
09:        var myVideo = document.getElementById("myVideo");
10:        myVideo.src = myVideoURL;
11:        myVideo.controls = true;
12:        myVideo.load();
13:        myVideo.play();
14:      }
15:    </script>
16:  </head>
17:  <body>
18:    輸入影片的網址：<input type="text" id="myVideoURL">
19:    <input type="button" value=" 播放 " onclick="playVideo()">
20:    <video id="myVideo"></video>
21:  </body>
22:</html>
```

<\ 附錄 C\video5.html>

- 18：在網頁上插入一個單行文字方塊供使用者輸入影片的網址。
- 19：在網頁上插入一個按鈕，上面顯示的文字為「播放」，同時透過 onclick 屬性設定當使用者按下此按鈕時，就去呼叫 playVideo() 函式，這是一個以 JavaScript 撰寫的函式，用來取得影片的網址，然後下載並播放影片。
- 07 ~ 14：定義 playVideo() 函式，其中第 08 行是將使用者輸入的網址存放在變數 myVideoURL，第 10 行是將 <video> 元素的 src 屬性設定為使用者輸入的網址，第 11 行是將 <video> 元素的 controls 屬性設定為 true，以顯示控制面板，第 12 行是呼叫 <video> 元素的 load() 方法下載影片，而第 13 行是呼叫 <video> 元素的 play() 方法播放影片。

C-2 <video> 與 <audio> 元素的事件

除了前述的屬性與方法，<video> 與 <audio> 元素還有事件，而且我們可以把這些事件發生的時間歸納成兩類，其一是讀取媒體資料時，其二是播放媒體資料時，下表為讀取媒體資料時所發生的事件 (參考自 HTML5 官方文件)。

讀取媒體資料時所發生的事件	說明
loadstart	開始尋找媒體資料，此時，networkState 屬性等於 NETWORK_LOADING。
progress	正在讀取媒體資料，此時，networkState 屬性等於 NETWORK_LOADING。
suspend	中斷讀取媒體資料，此時，networkState 屬性等於 NETWORK_IDLE。
abort	停止讀取媒體資料，但不是因為錯誤，此時，networkState 屬性等於 NETWORK_EMPTY 或 NETWORK_IDLE，視何時停止讀取而定。
error	讀取媒體資料時發生錯誤，此時，networkState 屬性等於 NETWORK_EMPTY 或 NETWORK_IDLE，視何時停止下載而定。
emptied	當媒體元素的 networkState 屬性從不是 NETWORK_EMPTY 切換到 NETWORK_EMPTY 的那一刻。
stalled	讀取媒體資料的速度變慢，此時，networkState 屬性等於 NETWORK_LOADING。
loadedmetadata	讀取媒體資料的 metadata，裡面可能包括播放長度、畫面寬度高度、字幕等資訊，此時，readyState 屬性等於 HAVE_METADATA 或比第一次大的值。
loadeddata	第一次可以從目前的播放位置演譯 (render) 媒體資料，此時，readyState 屬性等於 HAVE_CURRENT_DATA 或比第一次大的值。
canplay	可以開始播放，表示已經讀取某個程度的媒體資料，此時，readyState 屬性等於 HAVE_FUTURE_DATA 或更大的值。
canplaythrough	預估維持此下載速度的話，就能持續播放到結束，表示已經讀取某個程度的媒體資料，此時，readyState 屬性等於 HAVE_ENOUGH_DATA。

讀取媒體資料時所發生的事件具有一定的發生順序如下：

1. emptied
2. loadstart
3. loadedmetadata
4. loadeddata
5. canplay
6. canplaythrough

在此過程中，若讀取錯誤，會發生 error 事件；若中斷讀取，會發生 suspend 事件；若停止讀取，會發生 abort 事件；若讀取速度變慢，會發生 stalled 事件；而 progress 事件是只要在讀取中，就會持續發生。

至於播放媒體資料時所發生的事件則如下表 (參考自 HTML5 官方文件)。

播放媒體資料時所發生的事件	說明
play	最初呼叫 play() 方法或設定 aotuplay 屬性開始播放媒體資料，此時，paused 屬性等於 false。
playing	準備好開始播放 (之前可能被暫停或沒有媒體資料而導致延遲)，此時，readyState 屬性等於 HAVE_FUTURE_DATA 或更大的值。
timeupdate	目前的播放位置改變中，表示正在播放。
waiting	等待下一個畫格 (frame)，此時，readyState 屬性等於 HAVE_CURRENT_DATA 或更小的值，paused 屬性等於 false，seeking 屬性等於 true (表示需要尋找某個新的播放位置)。
seeking	seeking 屬性變為 true。
seeked	seeking 屬性變為 false。
volumechange	volume 或 muted 屬性有更新。

播放媒體資料時所發生的事件	說明
ended	因為抵達媒體資料的結尾而停止播放，即播放結束，此時，ended 屬性等於 true。
durationchange	duration 屬性的值有更新。
pause	媒體元素被暫停，在呼叫 pause() 方法時會發生 pause 事件，此時，paused 屬性等於 true。
ratechange	playbackRate 屬性有更新。

同樣的，播放媒體資料時所發生的事件亦具有一定的發生順序如下：

1. play

2. playing

3. timeupdate (只要在播放中，就會持續發生)

4. ended

我們來簡單示範如何使用 JavaScript 捕捉 <video> 元素的事件，然後加入自行撰寫的事件處理程式。下面的例子改寫自 <\ 附錄 C\video5.html>，它會捕捉 <video> 元素的 ended 事件，一旦捕捉到該事件，表示影片已經播放完畢，此時，就利用自行撰寫的事件處理程式在影片下方顯示影片播放時間。在充分瞭解其中的技巧後，您就可以試著舉一反三，捕捉其它事件並利用事件處理程式對您專屬的媒體播放程式做更多的調控。

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>影音多媒體</title>
```

<\ 附錄 C\video6.html>(下頁續 1/2)

```
06:     <script>
07:       function playVideo() {
08:         var myVideoURL = document.getElementById("myVideoURL").value;
09:         var myVideo = document.getElementById("myVideo");
10:         myVideo.src = myVideoURL;
11:         myVideo.controls = true;
12:         myVideo.load();
13:         myVideo.play();
14:         myVideo.addEventListener("ended", function() {
15:           var showTime = document.getElementById("showTime");
16:           showTime.innerHTML = "影片播放時間為" + myVideo.duration + "秒";
17:         }, false);
18:       }
19:     </script>
20:   </head>
21:   <body>
22:     輸入影片的網址：<input type="text" id="myVideoURL">
23:     <input type="button" value="播放" onclick="playVideo()">
24:     <video id="myVideo"></video>
25:     <p id="showTime"></p>
26:   </body>
27:</html>
```

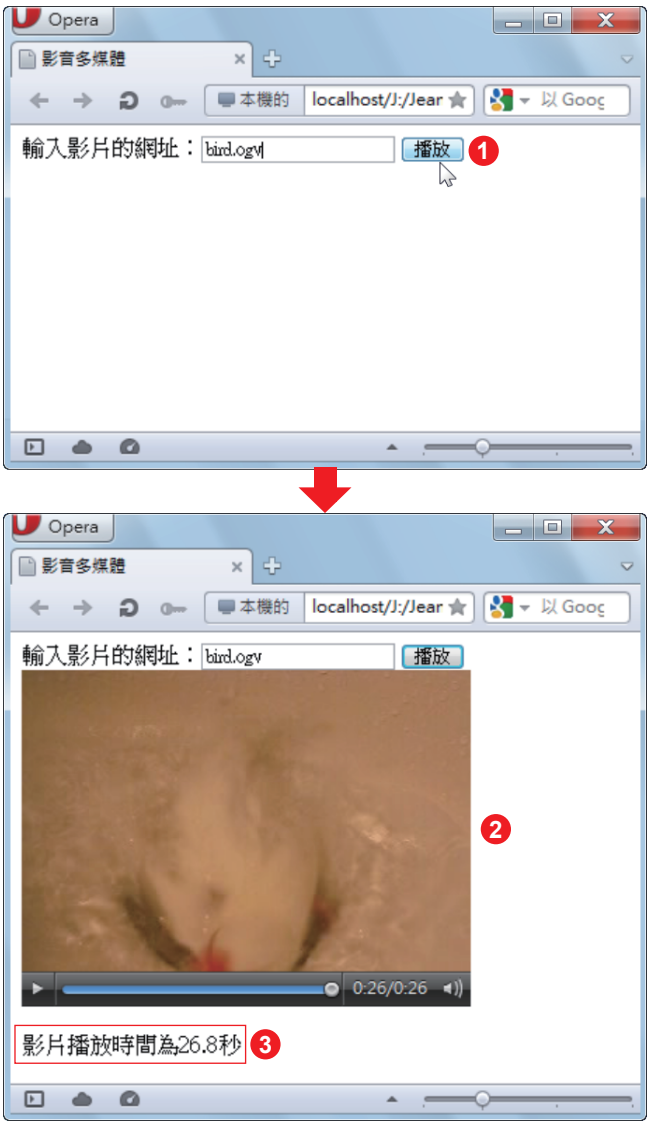
<\ 附錄 C\video6.htm> (接上頁 2/2)

❖ 25：插入一個段落，將其 id 屬性設定為 "showTime"，當影片播放完畢時，就會在此段落顯示影片播放時間。

❖ 14 ~ 17：利用 addEventListener() 方法捕捉 ended 事件並撰寫事件處理程式，其中第 16 行是將段落的內容設定為 <video> 元素的 duration 屬性，也就是影片的播放長度。提醒您，addEventListener() 方法的語法如下，參數 *event* 是要捕捉的事件，參數 *function* 是要執行的函式，而參數 *useCapture* 是布林值，通常為 false，表示當內層和外層元素都有發生參數 *event* 指定的事件時，先從內層元素開始執行處理程式：

```
addEventListener(event, function [, useCapture])
```

這個例子的瀏覽結果如下圖。



1 輸入影片的網址後按 [播放]　3 當影片播放完畢時會顯示播放時間

2 開始下載並播放影片