Appendix

Geolocation API

E

E-1 HTML5 的地理定位功能

「地理定位」是 HTML5 非常酷的一項新功能，它可以找出使用者的位置，然後與可信任的使用者代理程式分享位置資訊。W3C 針對地理定位功能提供了一組 Geolocation API，不過，Geolocation API 並沒有納入 W3C HTML5 的核心文件，而是單獨出版文件 (http://dev.w3.org/geo/api/spec-source.html)。

隨著智慧型手機、平板電腦等行動裝置日益普及，更凸顯出這項新功能的實用性，舉例來說，只要使用者允許瀏覽器取得其位置資訊，就可以在地圖上標示使用者的位置、提供距離使用者最近的門市或觀光景點、追蹤使用者的移動路線或移動距離等。

所謂「位置資訊」主要是指緯度 (latitude) 與經度 (longitude)，HTML5 是採用十進位格式表示緯度與經度，例如台北市內湖區的緯度約 25.093596、經度約 121.594077，而高雄市的緯度約 22.638095、經度約 120.325699。除了緯度與經度之外，HTML5 的地理定位功能還提供準確度 (accuracy)，這指的是所偵測的位置與實際位置的誤差範圍。

至於位置資訊是從哪來的呢？事實上，Geolocation API 並沒有指明裝置應該使用何種技術取得位置資訊，只是很單純的提供了下列三個方法 (後續的小節有進一步的說明)：

- void getCurrentPosition(PositionCallback *successCallback*, optional PositionErrorCallback *errorCallback*, optional PositionOptions *options*)：單次取得使用者的位置。
- long watchPosition(PositionCallback *successCallback*, optional PositionErrorCallback *errorCallback*, optional PositionOptions *options*)：持續追蹤使用者的位置。
- void clearWatch(long *watchId*)：取消追蹤使用者的位置。

裝置通常可以透過下列幾種來源取得位置資訊：

- IP 位址：透過 IP 位址取得位置資訊聽起來很直覺，卻不一定準確，因為所取得的位置資訊可能是提供 IP 位址給使用者，位於數公里或數十公里外的 ISP，而不是使用者真正的位置。

- GPS：這是利用裝置上的 GPS 晶片進行定位，誤差範圍可以縮小到幾公尺之內，優點是定位較準確，缺點則是定位速度較慢，因為 GPS 晶片較耗電，行動裝置通常會關閉 GPS 晶片，等到有需要的時候才開啟，所以在完成初始化到定位完畢，往往需要等待一段延遲時間。

  註：GPS (Global Positioning System，全球定位系統) 是由美國國防部所建置，該系統在 6 個軌道上使用 24 顆人造衛星，透過接收器接收並分析人造衛星傳回來的訊號，進而決定接收器的地理位置，以應用於地面與海上導航，目前有許多行動裝置內建 GPS 晶片。

- 行動電話基地台或無線上網熱點 (例如 Wi-Fi、藍牙)：這是根據使用者與行動電話基地台或無線上網熱點的距離，透過三角定位的方式來取得位置資訊，優點是定位速度較快，而且不需要配備精密的 GPS 晶片，缺點則是定位較粗略，誤差範圍可能是幾公尺或公里。

- 使用者輸入：與其讓裝置自行偵測位置，網頁提供一個介面讓使用者輸入地址或選擇所在的區域，也不失為一個好主意，這樣就不用擔心誤差範圍太大或延遲時間太久，現在有不少餐廳或商店的網站就是讓使用者先選擇所在的區域，然後再顯示該區域內的分店資訊。

E-2 使用 Geolocation API

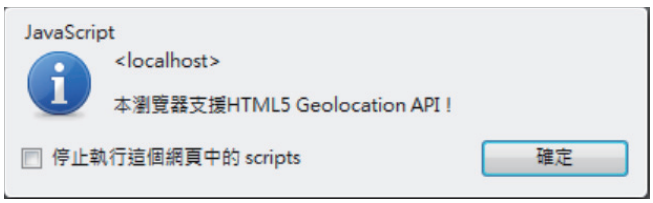
E-2-1 測試瀏覽器的地理定位功能

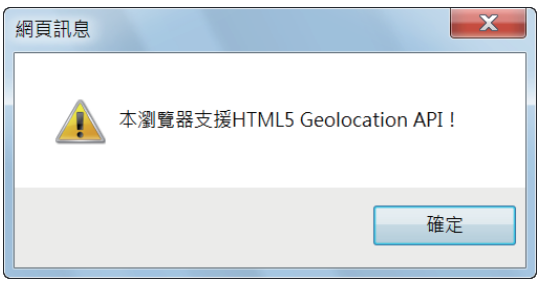
地理定位功能雖然是新穎的技術，但諸如 Opera 10.5+、Chrome 3.0+、Safari 4.0+、FireFox 3.5+、Internet Explorer 9 等瀏覽器均已經支援 Geolocation API，我們可以透過類似如下的程式碼測試瀏覽器是否支援 Geolocation API 。

```
<script>
  // 若 navigator.geolocation 物件存在，就表示支援 Geolocation API，否則表示不支援
  if (navigator.geolocation) {
    alert("本瀏覽器支援 HTML5 Geolocation API ! ");
  }
  else {
    alert("本瀏覽器不支援 HTML5 Geolocation API ! ");
  }
</script>
```

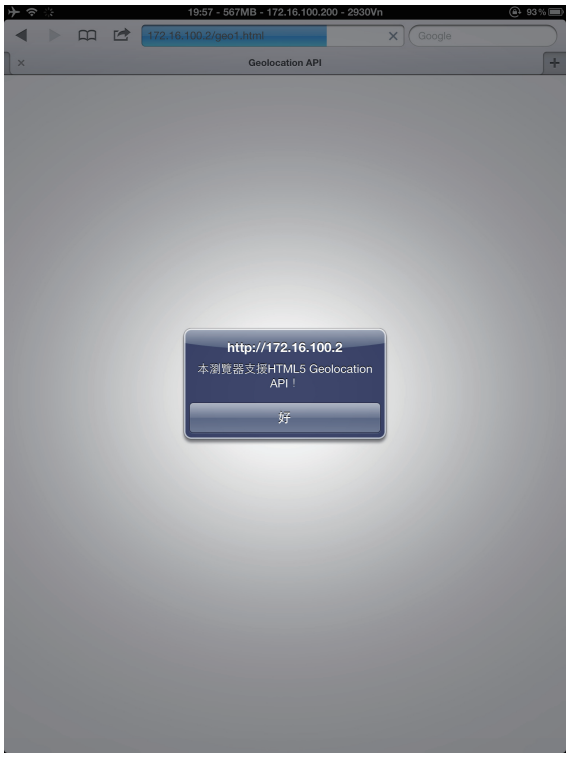
<\ 附錄 E\geo0.html>

這段程式碼在 Opera 與 Internet Explorer 9 執行會分別出現如下圖的對話方塊，表示支援 Geolocation API。





行動裝置上的瀏覽器對於 Geolocation API 的支援程度亦相當不錯，前述的程式碼在 iPad 與 iPhone 執行會分別出現如下圖的對話方塊，表示支援 Geolocation API。



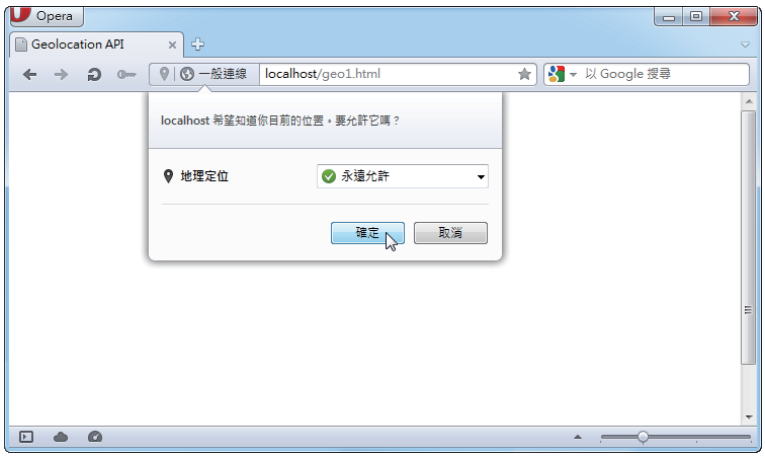
E-2-2 單次取得使用者的位置

Geolocation API 總共就只有三個方法－ getCurrentPosition()、watchPosition()、clearWatch()，getCurrentPosition() 用來單次取得使用者的位置，其語法如下：

```
void getCurrentPosition(PositionCallback successCallback, optional PositionErrorCallback
  errorCallback, optional PositionOptions options)
```

- *successCallback*：這個參數用來告訴瀏覽器在取得位置資訊成功時，應該呼叫哪個函式。
- *errorCallback*：這個選擇性參數用來告訴瀏覽器在取得位置資訊失敗時，應該呼叫哪個函式。
- *options*：這個選擇性參數用來提供一個 options 物件給地理定位服務，以指定蒐集資料的選項，例如設定等候逾時的時間。

基於隱私權的考量，HTML5 規格書中有提到：「在使用者表示允許之前，使用者代理程式不得將位置資訊傳送給網站」，因此，當使用者第一次連線到使用 Geolocation API 的網站時，瀏覽器必須清楚的詢問使用者是否允許地理定位功能，除非使用者表示允許，才能繼續後面的動作，以下各圖分別是 Opera、Internet Explorer 9、iPhone Safari 等瀏覽器的詢問畫面。



請注意，有些瀏覽器可以讓您選擇永遠允許或允許一次，為了保護隱私，建議您選擇允許一次。

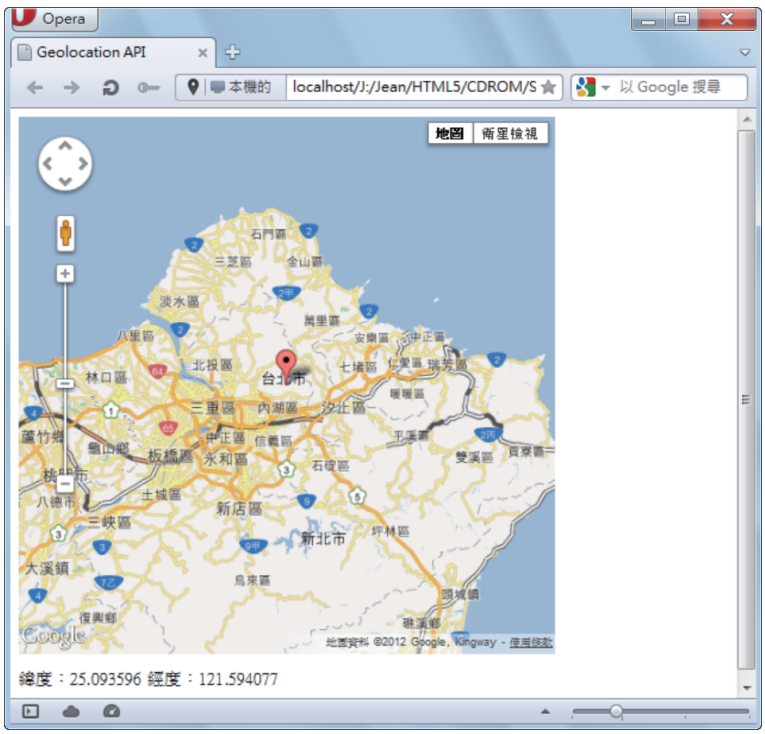
getCurrentPosition() 方法的第一個參數

getCurrentPosition() 方法的第一個參數用來告訴瀏覽器在取得位置資訊成功時，應該呼叫哪個函式，例如下面的敘述是指定要呼叫 geoSuccess() 函式：

```
navigator.geolocation.getCurrentPosition(geoSuccess);
```

至於 geoSuccess() 函式要做什麼，就視實際的應用而定，比方說，在 Google Maps 上標示使用者的位置、從資料庫中查詢距離使用者最近的門市或觀光景點、追蹤使用者的移動路線等。

下面是一個例子，它會呼叫 getCurrentPosition() 方法取得使用者的位置，然後在 Google Maps 上標示該位置並顯示其緯度與經度，下圖分別是 Opera 與 iPhone Safari 的執行畫面。




```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>Geolocation API</title>
06:    <script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=false">
07:    </script>
08:    <script>
09:      // 呼叫 getCurrentPosition() 方法取得使用者的位置
10:      navigator.geolocation.getCurrentPosition(geoSuccess);
11:
12:      // 在取得位置資訊成功時，呼叫 geoSuccess() 函式
13:      function geoSuccess(position){
14:        var geocoder = new google.maps.Geocoder();
15:        var latlng = new google.maps.LatLng(position.coords.latitude, position.coords.longitude);
```

<\ 附錄 *E\geo1.html>(* 下頁續 *1/2)*

```
16:          var myOptions = {zoom:10, center:latlng, mapTypeId:google.maps.MapTypeId.ROADMAP};
17:          var map = new google.maps.Map(document.getElementById("map_canvas"), myOptions);
18:          var marker = new google.maps.Marker({map:map, position:latlng});
19:          document.getElementById("msg").innerHTML = "緯度：" + position.coords.latitude +
                                        "\t 經度：" + position.coords.longitude;
20:       }
21:     </script>
22:   </head>
23:   <body>
24:     <div id="map_canvas" style="width:500px; height:500px;"></div>
25:     <p id="msg"></p>
26:   </body>
27:</html>
```

<\ 附錄 *E\geo1.html>(* 接上頁 *2/2)*

- 06、07：載入 Google Maps API，這些 API 是以 JavaScript 所撰寫。當您使用 Google Maps API 時，必須透過 sensor 參數指明應用程式是否有使用感應器 (例如 GPS 定位器) 來判斷使用者的位置，此處因為沒有使用感應器，所以是傳送 sensor=false。

- 10：呼叫 getCurrentPosition() 方法取得使用者的位置，此處只有傳入一個參數 geoSuccess，表示在取得位置資訊成功時，呼叫 geoSuccess() 函式。

- 13 ~ 20：定義 geoSuccess() 函式，它會在 Google Maps 上標示使用者的位置並顯示其緯度與經度。

  請注意，geoSuccess() 函式只有一個參數，這是一個 position 物件，包含使用者的位置資訊。position 物件有兩個屬性－ timestamp 與 coords，timestamp 是取得位置資訊的時間 (從 1970/1/1 開始所經過的毫秒數)，而 coords 是進一步的位置資訊，包含下列的唯讀屬性：

  - latitude：緯度 (採用十進位格式)。
  - longitude：經度 (採用十進位格式)。

- accuracy：準確度 (以公尺為單位)，這指的是所偵測的位置與實際位置的誤差範圍，建議您在取得位置資訊時可以檢查 accuracy 屬性的值，若誤差範圍太大，就改成讓使用者輸入地址或選擇所在的區域。
- altitude：海拔高度 (以公尺為單位)。
- altitudeAccuracy：海拔高度的準確度 (以公尺為單位)。
- heading：裝置的行進方向，以面向正北方順時針方向的角度來表示，即 0°≤ heading < 360°，若裝置是靜止不動的 (即 speed 屬性為 0)，則其值為 NaN。
- speed：裝置的行進速度 (以公尺 / 秒為單位)。

瀏覽器不一定有支援 altitude、altitudeAccuracy、heading、speed 等屬性，若瀏覽器尚未支援這些屬性，則其值為 null。

❖ 14：建立一個隸屬於 google.maps.Geocoder 類別的物件，然後指派給變數 geocoder，用來存放地理編碼的結果，所謂「地理編碼」就是將地址轉換為地理座標，例如高雄市可以轉換為緯度 22.638095、經度 120.325699。

❖ 15：建立一個隸屬於 google.maps.LatLng 類別的物件，然後指派給變數 latlng，用來存放緯度與經度，此處是將預設的緯度與經度指派為第 10 行所取得的位置資訊，即 position.coords.latitude 與 position.coords.longitude。

❖ 16：定義一個變數 myOptiopn，用來指定地圖選項，包括縮放比例為 10、中心點為變數 latlng 所指定的緯度與經度、地圖類型為預設的 2D 地圖。

❖ 17：建立一個隸屬於 google.maps.Map 類別的物件，然後指派給變數 map，用來存放地圖，而且該地圖會顯示在 HTML 網頁中 id 屬性為 "map_canvas" 的元素內，此處指的是第 24 行所插入的 ... 區塊。

❖ 18：建立一個 marker 在 Google Maps 標示第 10 行所取得的位置。

❖ 19：在地圖下面的段落顯示位置資訊的緯度與經度。

getCurrentPosition() 方法的第二個參數

在前面的例子中，我們並沒有在 getCurrentPosition() 方法傳入第二個參數，這個選擇性參數用來告訴瀏覽器在取得位置資訊失敗時，應該呼叫哪個函式，例如下面的敘述是指定要呼叫 geoError() 函式：

```
navigator.geolocation.getCurrentPosition(geoSuccess, geoError);
```

建議您在此處撰寫錯誤處理函式，因為瀏覽器不見得每次都能取得位置資訊成功，可能是使用者不允許存取其位置，也可能是裝置剛好收不到訊號或沒電。

錯誤處理函式只有一個參數，這是一個 error 物件，包含錯誤資訊。error 物件有兩個屬性－ message 與 code，message 是錯誤訊息，即描述發生錯誤的原因，而 code 是錯誤碼，其值有下列幾種：

- PERMISSION_DENIED (數值 1)：使用者不允許存取其位置。
- POSITION_UNAVAILABLE (數值 2)：無法取得使用者的位置。
- TIMEOUT (數值 3)：等候逾時。

例如下面的錯誤處理函式會在取得位置資訊失敗時，顯示發生錯誤的原因：

```
function geoError(error){
  switch (error.code) {
    case 1:
      alert("使用者不允許存取其位置");
      break;
    case 2:
      alert("無法取得使用者的位置");
      break;
    case 3:
      alert("等候逾時");
      break;
  }
}
```

getCurrentPosition() 方法的第三個參數

getCurrentPosition() 方法的第三個參數是一個選擇性參數，用來提供一個 options 物件給地理定位服務，以指定蒐集資料的選項，包括：

- enableHighAccuracy：告訴瀏覽器啟用高準確度模式提供地理定位服務，此選項預設為 false，表示不啟用。請謹慎設定 enableHighAccuracy 選項，有時就算將它設定為 true，也不一定比較準確，但卻比較費時或耗電，因為可能要啟用裝置上的 GPS 晶片進行定位。

- timeout：設定等候逾時的時間 (以毫秒為單位)，一旦瀏覽器超過指定的時間尚未取得位置資訊，就呼叫錯誤處理函式。此選項預設為 Infinity 或 0，表示無時間限制。

- maximumAge：設定位置資訊的有效時間 (以毫秒為單位)，一旦超過有效時間，瀏覽器就必須捨棄舊的位置資訊並試著去取得新的位置資訊。此選項預設為 0，表示瀏覽器每次發出新的要求，都必須去取得新的位置資訊。

例如下面的敘述是利用第三個參數加入 timeout:15000 選項，將等候逾時的時間設定為 15 秒鐘，一旦瀏覽器超過 15 秒鐘尚未取得位置資訊，就呼叫錯誤處理函式，此時 error.code (錯誤碼) 的值會被設定為 TIMEOUT：

```
navigator.geolocation.getCurrentPosition(geoSuccess, geoError, {timeout:15000});
```

至於下面的敘述則是比前一個敘述多加入 maximumAge:300000 選項，將位置資訊的有效時間設定為 5 分鐘，一旦超過 5 分鐘，瀏覽器就必須試著去取得新的位置資訊：

```
navigator.geolocation.getCurrentPosition(geoSuccess, geoError, {
  timeout:15000,
  maximumAge:300000
});
```

E-2-3 持續追蹤使用者的位置與取消追蹤

對於某些網頁應用程式來說，單次取得使用者的位置是不夠的，例如要在地圖上持續標示使用者的移動路徑、累計移動距離等，此時可以使用 Geolocation API 提供的 watchPosition() 方法持續追蹤使用者的位置，其語法如下：

```
long watchPosition(PositionCallback successCallback, optional PositionErrorCallback errorCallback,
    optional PositionOptions options)
```

watchPosition() 方法的參數和 getCurrentPosition() 方法相同，此處不再重複講解，要注意的是 watchPosition() 方法有傳回值，該值可以當作參數傳遞給 clearWatch() 方法，以取消追蹤，clearWatch() 方法的語法如下：

```
void clearWatch(long watchId)
```

下面是一個例子。

```
<script>
  // 呼叫 watchPosition() 方法持續追蹤使用者的位置
  var watchId = navigator.geolocation.watchPosition(geoSuccess);

  // 在成功取得位置資訊時，呼叫 geoSuccess() 函式，此例是顯示緯度與經度
  function geoSuccess(position){
    document.getElementById("msg").innerHTML = "緯度：" + position.coords.latitude +
                                                "\t 經度：" + position.coords.longitude;
  }

  ...

  // 呼叫 clearWatch() 方法取消追蹤使用者的位置
  navigator.geolocation.clearWatch(watchId);
</script>
```