

jQuery的安裝





手動下載安裝

下載 jQuery 後讀入

- 網址：<https://jquery.com/download/>
- `<script src= "jquery.js" ></script>`



jQuery slim

在 v3 後多了 slim 的版本，主要是拿掉了 ajax 和 effect 的部分，可以視情況是否要使用精簡版本的 jQuery。

使用cdn的方式安裝

使用 cdn

- <https://code.jquery.com/>

jQuery Core

Showing the latest stable release in each major branch. [See all versions of jQuery Core.](#)

jQuery 3.x

- jQuery Core 3.5.1 - [uncompressed](#), [minified](#), [slim](#), [slim minified](#)



CodePen 上使用 jQuery



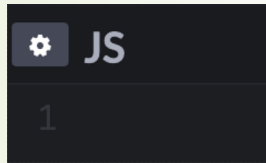
CodePen 介紹

CodePen

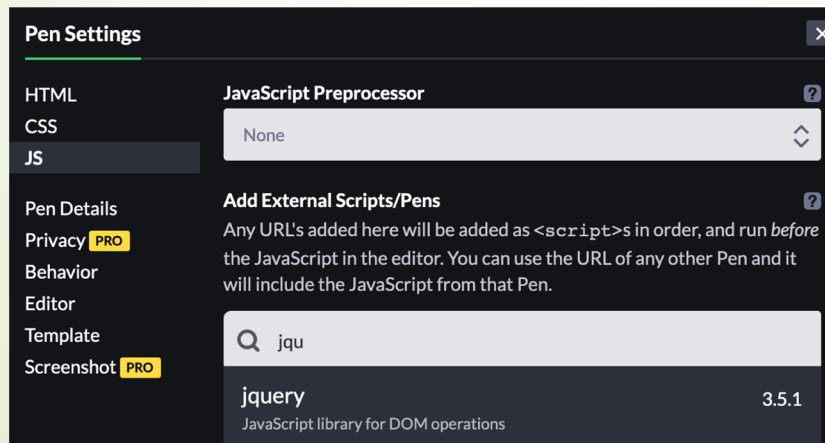
- <https://codepen.io>
- CodePen 是一個可以用來讓使用者練習及分享 code 的線上資源網站。

在 CodePen 使用 jQuery 的步驟與方法

1. 點擊 Pen 建立新的專案
2. 在 JS 的面板點擊設定按鈕



3. 在外部資源搜尋並選擇 jQuery





取得内容、属性

jQuery 基本用法

- `$(selector).action();`
- 使用 \$ 與 () 包著 CSS 的選擇器，就可以指定要控制的目標。



HTML 操作

`.text();`

`.html();`

`.attr();`

`.prop()`

`.val();`



`.text()`

取得 HTML tag 內的文字

取值：`.text()`




.html()

- 跟 .text() 類似，但會取得或設定 HTML tag 的 HTML 內容
- 取值：.html()



.attr()

- 取得 HTML tag **attribute** 的值
- ``
- 取值 `.attr(attributeName)`



更新屬性、取代內容



.text()

- 設定 HTML tag 內的文字
- 設值：.text(text)




.html()

- 設定 HTML tag 的 HTML 內容
- 設值：.html(html)



.attr()

- 設定 HTML tag **attribute** 的值
- ``
- 設值 `.attr(attributeName, value)`



移除元素、新增元素



插入 html 內容

- .append()- 插入到選擇的 DOM 末端
- .appendTo()- 將指定內容移動到指定 DOM 末端
- .prepend()- 插入到選擇的 DOM 前端
- .prependTo()- 將指定內容移動到指定 DOM 前端

插入到指定的 html 元素之後

.after(element)

- \$('選擇器').after('<div>After</div>');

.insertAfter(siblingElement)

- \$('<div>After</div>').insertAfter('選擇器');

插入指定 html 元素之前

.before(element)

`$('選擇器').before('<div> Before </div>');`

.insertBefore(siblingElement)

- `$('<div> Before </div>').insertBefore('選擇器');`



移除 html

`.empty()`- 清空選取之 DOM

`.remove()`- 刪除選取之 DOM



認識 選擇器 (name 、 id 、 class)



認識 name 、 id 、 class

name: div, p, span...

class: <div **class**='class'> </div>

id: <div **id**='id'> </div>



javascript selector

- `Document.getElementById("xyz")`
- `document.getElementsByTagName("p")`
- `document.getElementsByClassName("abc")`
- `document.querySelectorAll("span.a, span.c")`
- `document.querySelector("#id, .class")`

jQuery selector

- ➡ 類似 CSS 選擇器，差別在需要加上 `$()`
- ➡ 標籤選擇器 `button : $('button')`
- ➡ • class 選擇器: `.btn : $('.btn')`
- ➡ • id 選擇器: `#btn : $('#btn')`
- ➡ 屬性選擇器
 - `a[target='_blank'] : $('a[target='_blank'])`
- ➡ • 選擇多個 `.btn, .btns : $('.btn, .btns')`



Basic filter



Basic filter

:eq()- 選取第 n 個(從 0 開始)

:even- 選取奇數個

:odd- 選取偶數個

:first- 選取第一個

:last- 選取最後一個

:gt()- 選取大於第 n 個的(也可以給負數)

:lt()- 選取小於第 n 個的(也可以給負數)

:header- 選取 h1~h6



content filter

:contains()- 選取內容包含 xx

```
<div class="AAA">Jason Kidd</div>
```

```
<div class="AAA">LeBron James</div>
```

```
<div class="AAA">Jason Williams</div>
```

```
<div class="AAA">James Harden</div>
```

```
$("div.AAA:contains('Jason')").css("background", "yellow");
```



篩選(filter)函式

.first() 、 .last() 、 .eq() 函數



Filtering

- `.eq()`- 選取第 n 個
- `.first()`- 選取第一個
- `.last()`- 選取最後一個



篩選(filter)函式

.slice() 、 .has()



.slice()

- .slice(start, [end])- 選取特定範圍內的元素

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
<div> </div>
```

```
$( 'div' ).slice( 3, 8 ).css( 'background',  
'yellow' );
```



.has()

- .has()- 選取含有 xx 的元素

```
<div>
```

```
  <div class="a1">A1</div>
```

```
</div>
```

```
<div>
```

```
  <div class="a2">A2</div>
```

```
</div>
```

```
$("#div").has(".a1").css("background", "red");
```



.not()

.not()- 選取不是 xx 的元素

```
<div class="a1">A1</div>
```

```
<div class="a2">A2</div>
```

```
<div class="a3">A3</div>
```

```
$("div").not(".a1").css("background", "red");
```



.filter()

- .filter(selector)- 進一步在選取範圍再做篩選

```
<div> </div>  
<div class="middle"> </div>  
<div class="middle"> </div>  
<div class="middle"> </div>  
<div class="middle"> </div>  
<div> </div>
```

```
$( "div" )  
.css( "background", "#c8ebcc" )  
.filter( ".middle" )  
.css( "border-color", "red" );
```



尋訪(Traversing)函式

`.prev()` 、 `.next()`



.next()

.next()- 往後找一個元素

.nextAll()- 往後及其之後的元素

.nextUntil()- 往後選但到特定元素後停止



.prev()

.prev()- 往前找一個元素

.prevAll()- 往前及其之前的元素

.prevUntil()- 往前選但到特定元素停止



尋訪(Traversing)函式

`.siblings()` 、 `.parent()` 、 `.parents()`



.siblings()

.siblings()- 自己以外的元素 (兄弟姐妹)

```
<div class="a1">A1</div>
```

```
<div class="a2">A2</div>
```

```
<div class="a3">A3</div>
```

```
<div class="a4">A4</div>
```

```
<div class="a5">A5</div>
```

```
$(".a2").siblings().css("background", "red");
```



.parent()

.parent()- 往上選取一層（父層）

.closest()- 最接近 祖先元素


.parents()- 往上選取全部

.parentsUntil()- 往上選取全部並設定停止的元素



尋訪(Traversing)函式

`.children()`
`.find()`



`.children()` 與 `.find()`

`.children()`- 所有條件相符子元素（只找一層）

`.find()`- 找條件相符子孫元素



.end()

- .end()- 回到上一個選取狀態

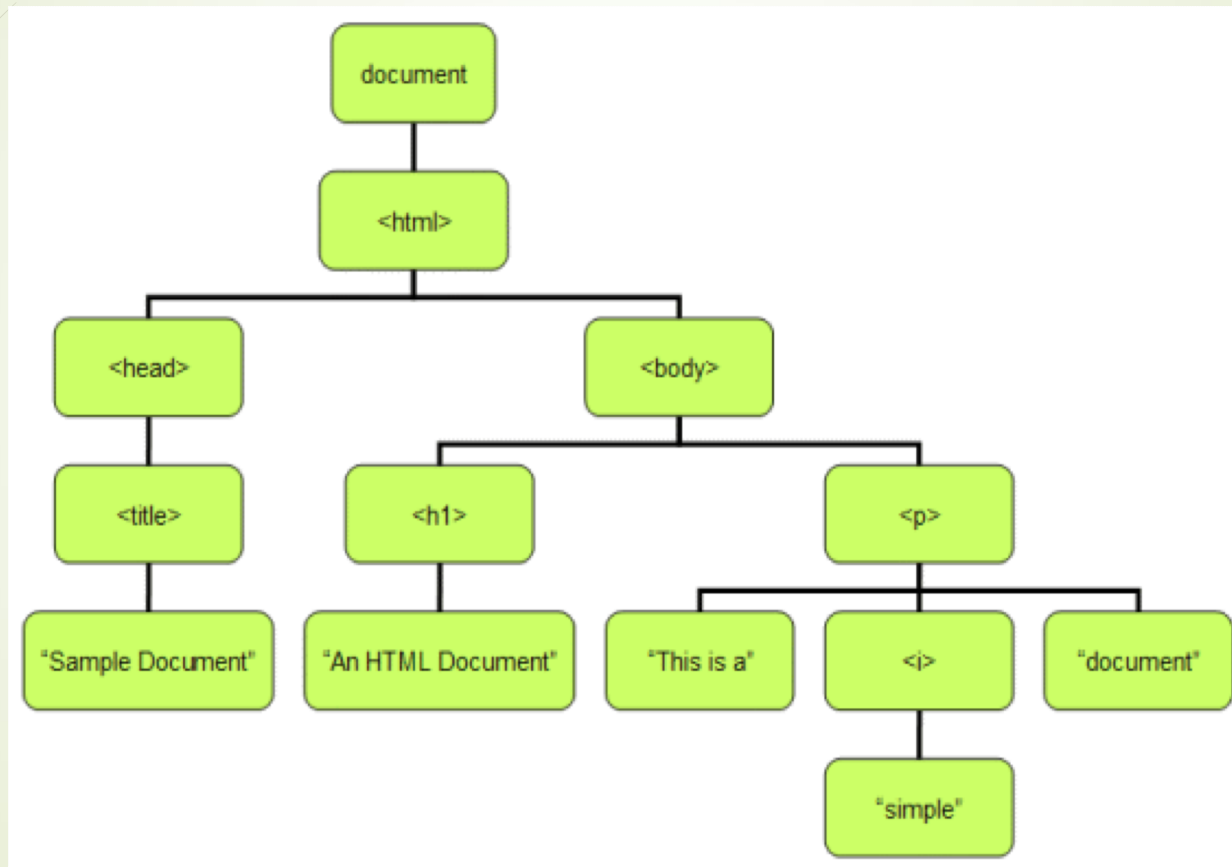
```
<ul class="first">
  <li class=" a1">list item 1</li>
  <li>list item 2</li>
  <li class=" a2">list item 3</li>
</ul>
<ul class="second">
  <li class=" a1">list item 1</li>
  <li>list item 2</li>
  <li class=" 2">list item 3</li>
</ul>
```

```
$( "ul.first" )
  .find( ".a1" )
    .css( "background-color", "red" )
  .end()
  .find( ".a2" )
    .css( "background-color", "green"
);
```



DOM -1

文件物件模型 (DOM)



出處：<https://www.cs.pu.edu.tw/~tsay/course/webprog/notes/DOM>

DOM屬性的操控

.attr()

- 設定 HTML tag **attribute** 的值
 - ``
- 設值 `.attr(attributeName, value)`



.attr()

使用物件方式：

```
.attr({  
    屬性1: 值1,  
    屬性2: 值2  
})
```



樣式的操控



.css()

.css('css 敘述')- 取得值

.css('css 敘述', '值')- 設定值

設定多個樣式

```
let styles = { 敘述1: '值' ,  
              敘述2: '值'  
              };  
$( this ).css( styles );
```



.css()

累進値：

ex : `css('left', '+=2');`

取得區塊設定的背景顏色

- 用 `.css('background-color')` 取得顏色後輸出



the color is `rgb(255, 0, 0)`

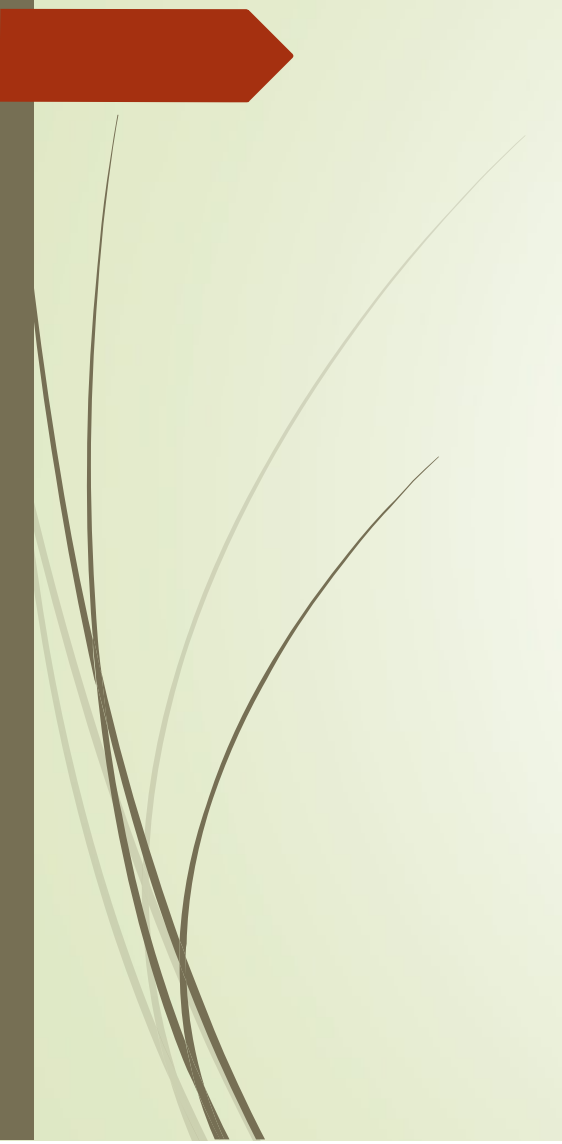


DOM-2



.html()

- 設定 HTML tag 的 HTML 內容
- 設值：.html(html)



取得元素的位置座標

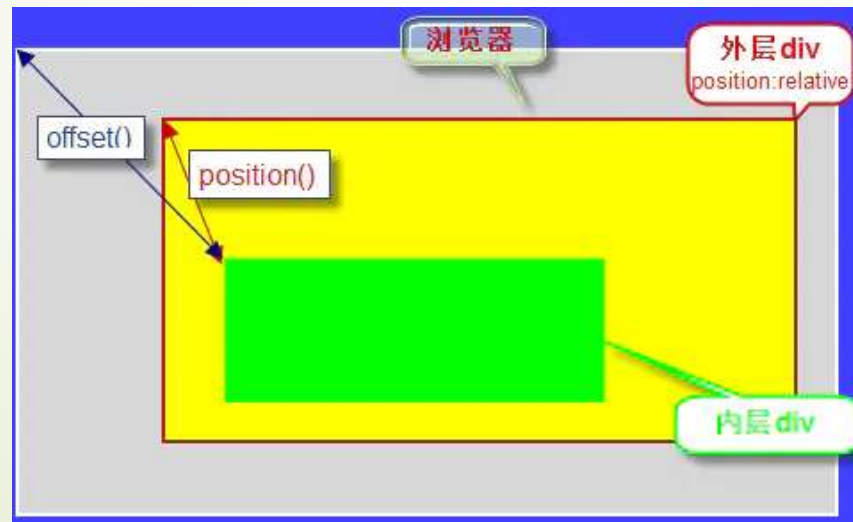


`.position()`

- `$(selector).position()` 取得位置

.position() 與.offset()差異

- position() 取得 離他最近有 position:relative 屬性父元素相對位置
- .offset() 取得當前 頁面上的座標位置 (絕對位置)





.clone() 複製元素

- ➡ 把內容複製到指定的地方



表單相關的操控



表單選擇器 (Form selector)

- :button(input, button)
- :checkbox
- :checked
- :disabled
- :enabled
- :file
- :focus
- :image

- :input
- :password
- :radio
- :reset
- :selected
- :submit
- :text



表單事件 (Form Event)

- .focus()-
當通過鼠標點擊選中元素或通過tab 鍵定位到元素時，該元素就會獲得焦點
- .blur()
當元素失去焦點時發生blur 事件
(先有.focus() 才會有.blur())
- .change()- 偵測 input 或 select 變化
- .submit()- 送出表單



.val()

- 取得或設定表單元素 (input, select...) 的值
- 取值 .val()
- 給值 .val(value)



.prop()

- 類似 .attr()，但只有 **true** 跟 **false** 的值
 - 建議用來設定或 取得如checked 或 disabled 只有兩種狀態的 attribute 。
- 取值：.prop(propertyName)
- 設值：.prop(propertyName, value)



文字的操控



.text()

- 取得 HTML tag 內的文字
- 取值：.text()



.text()

- 設定 HTML tag 內的文字
- 設值：.text(text)



HTML5

自定義屬性 data- 的操控

.data()

- HTML5 後新增自訂屬性 data-*, 於html中可以紀錄一些資訊
jQuery 可以直接取得這些資訊
- 例 :

HTML

```
<div class="text-center" id="nameCard" data-name="Jason" data-  
last-name="Kidd" data-age="42">Jason Kidd</div>
```

jQuery 取值

```
$("#nameCard").data('name')
```

```
$("#nameCard").data('lastName')
```

```
$("#nameCard").data('age')
```



事件的介紹與 綁定寫法



jQuery Event

- Mouse event
- Keyboard event
- Form event
- Document/Window event



滑鼠事件 (Mouse Event)


- .click()- 點擊
- .dblclick()- 連點
- .contextmenu()- 右鍵
- .mousedown()- 滑鼠按下
- .mouseup()- 離開滑鼠
- .mouseenter()- 滑鼠移進選擇器
- .mouseleave()- 滑鼠離開
- .hover()- 移進與離開



.hover()

hover 滑進滑出都會回傳事件，
可以別觸發之

```
.hover(function(){  
    //偵測滑入  
}, function(){  
    //偵測滑出  
});
```



認識 DOMContentLoaded 與 load 事件



DOMContentLoaded

DOMContentLoaded 事件是當 document 被完整的讀取跟解析後就會被觸發(不會等待 stylesheets，圖片和 subframes 完成讀取)。

DOM ready

因為 jQuery 是把事件綁到 DOM 上，用這種寫法會等頁面上的 DOM 讀取完後才執行程式。

```
$(document).ready(function(){  
  
});
```

或

```
$(function(){  
});
```




load

相較於 DOMContentLoaded 事件，
load 的事件則是確定頁面的圖片與樣式都
已讀取完後才觸發。



.load()



```
$("#img").load(function(){  
    alert("Image loaded.");  
});
```



認識事件物件 (Event Object)

Event Object

- 事件(event)在瀏覽器中是以物件的形式存在的
- 觸發一個事件物件
 - 該物件包含著所有與事件有關的資訊。包括導致事件的元素、事件的型別以及其他特定事件相關的資訊。

```
$( ".btn" ).click(function(event){  
})
```




Event Object

- 使用者在瀏覽網頁的時候會觸發很多事件（Event），例如按下滑鼠或按下鍵盤等
- DOM Event 幫我們定義了很多種事件型態，讓我們可以用 JavaScript 來監聽（listen）和處理（event handling）這些事件。

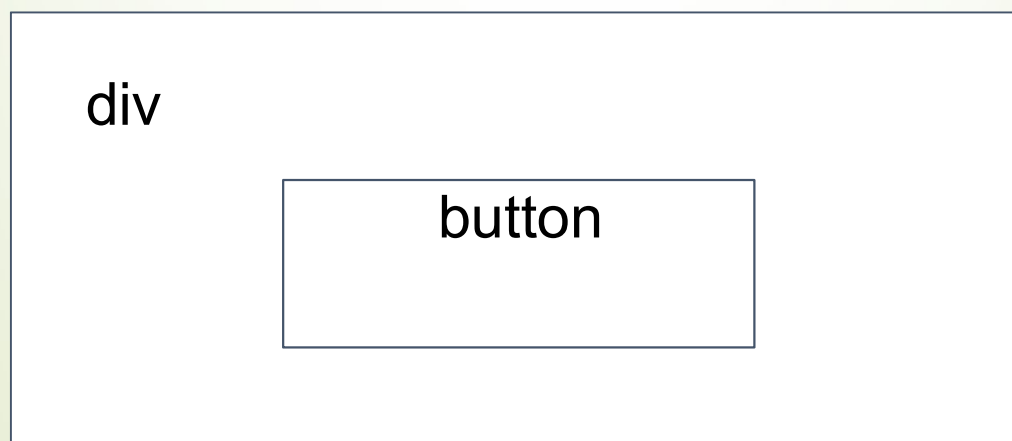



事件觸發寫法及 注意事項

(例：event bubble 事件冒泡狀況)

Event bubble(事件冒泡)

- 當元素被巢狀結構大包小的時候，觸發裡層的元素的事件，同時也會觸發 外層元素的同一事件。
- 例如當我們點擊下圖的 button，也會觸發 div 的點擊事件。





阻止事件冒泡

利用 `event.stopPropagation()` 來
阻止事件的冒泡傳遞。



動態事件的綁定



.On()

- Selector 是後來才動態產生事件的綁定
- 一個 selector 需要綁定多個事件時
- 不同事件觸發一樣的動作時

selector 動態產生事件的綁定

- 因為 jQuery 會在 HTML 畫完後根據既有的結構將事件綁到 HTML 上，若是靠 程式事後產生的話，原本寫的就無法綁到 HTML 上。
- `$(選擇器(原本就存在 HTML 上之元素)).on(觸發方式(原本的 " click" , " mouseenter" ...等) [, 靠程式動態產生的元素] , 事件處理 (function(){}))`

selector 動態產生事件的綁定

➤ .on(events [, selector] [, data], handler)

➤ 例1

```
$( "#clickMe" ).on( "click", function() {  
    console.log( $( this ).text() );  
});
```

➤ 例2

```
$( "ul" ).on( "click", "li", function() {  
    console.log( $( this ).text() );  
});
```




事件的觸發 與解除綁定



一個 selector 綁定多個事件 (物件的表示方式)

```
$("#button").on({  
    mouseenter: function(){  
        $("#showAction").text("Mouse Enter");  
    },  
    click: function(){  
        $("#showAction").text("Click");  
    },  
    dblclick: function(){  
        $("#showAction").text("Double Click");  
    }  
});
```



不同事件觸發同一個動作時

```
$("#button").on("click mouseenter ...",  
function(){  
    //do something  
});
```



.off()


- 取消由 on 建立的事件



.one()

只執行一次

.one(events [, data], handler)



動畫特效

- 動畫特效是原本 jquery 的強項，但在 css3 的動畫普及後，重要性就大幅降低。
- 動畫特效的部分記得要不能讀取到 jquery.slim.js。
- 尤其是使用 bootstrap 的話，預設會讀取 jquery.slim.js。




Basic

`.show()`

`.hide()`

`.toggle()`

於 `.show()` 與 `.hide()` 之間切換



Fade Effect(淡入淡出)

.fadeIn()

.fadeOut()

.fadeToggle()



Slide Effect(滑入滑出)

`.slideDown()`

`.slideUp()`

`.slideToggle()`



增添網頁動畫豐富度

Animate.css 外掛

- Animate.css : <https://animate.style/>
- CDN

```
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/animate
.css/4.1.1/animate.min.css"
/>
```



`.animate()` 中的 CSS 設定與時間設定



.animate()

- .animate(properties [, duration] [, easing] [, complete])
- duration: 單位- ms(1/1000 s)

```
$(".box").animate({  
    left: "+=50" },  
    500, function(){  
        //after animation  
    });
```

easing

- 控制動畫過程中的加減速效果。
- 原生的 timing function 很少，需要靠 plugin 補強。
- jQuery easing
 - 外掛位置
 - <http://gsgd.co.uk/sandbox/jquery/easing/>



.stop()

- 停止動畫
- `$(selector).stop(stopAll,goToEnd)`



.delay()

- 動畫延遲
- 單位: ms(1/1000 s)
- 除了 ms 外還可以用 'fast' 跟 'slow'



實用函式一1



判斷是否為陣列

語法：\$.isArray()

`Array.isArray([1, 2, 3]); // true`

`Array.isArray({foo: 123}); // false`

`Array.isArray('foobar'); // false`




`$.isArray()`

判斷是否有該值：

檢查陣列是否有某特定值

- ➡ 語法：`$.isArray()`
- ➡ `$.isArray(value, array [, fromIndex])`
 - ➡ 無：回傳 -1
 - ➡ 有：回傳 註標值



陣列的合併：\$.merge()


語法：\$.merge(first, second)

\$.merge([0, 1, 2], [2, 3, 4])



`$.extend()`

物件的合併與取代



`$.extend(target, object1 [, objectN])`

```
var object1 = { apple: 0,  
  banana: { weight: 52, price: 100 }, cherry: 97  
};  
var object2 = { banana: { price: 200 }, durian:  
  100  
};  
// Merge object2 into object1  
$.extend( object1, object2 );
```



\$.each() :

執行陣列、物件迴圈




.each() :


執行陣列、物件迴圈

jQuery 比較簡單的處理迴圈的方法，效能比較差但用起來比較快，用來處理 html 內容很方便

```
$("#table  
    td").each(function(){  
    console.log($(this).tex  
t());  
});
```

\$.map()
執行陣列、物件迴圈，
產生新的陣列或物件



```
$.map( array, function(element, index ){  
    return 新陣列的元素  
});
```

```
let arr = [ 1, 2, 3, 4, 5, 6];  
arr = $.map(arr, function(n, i)  
    {  
        return (n+1);  
    });
```

jQuery基本套件



建立自己的 jQuery Plugin (外掛)

先確保\$是給jQuery使用的

```
(function($) {
```

```
})(jQuery);
```

例：color.js



自訂名稱

```
$.fn.my = function(){  
}
```




簡單寫一個修改 css 的功能

```
$.fn.my = function(){  
    this.css( "color", "red" );  
    return this;  
}
```

//這樣就可以使用了

```
$(elem).my();
```



使用

```
$(elem).my({  
    color: "green" ;  
    backgroundColor:  
        "lightblue"  
});
```



套用別人寫好的plug in



載入第三方Plugin細節

- 確認熱門瀏覽器有無支援
- 確保該載入的CSS、JS、IMG都有載入
- 如果發現有問題時，打開console確認是否有錯誤
- 查詢外掛的JS設定
- 修改CSS成自己要的樣式