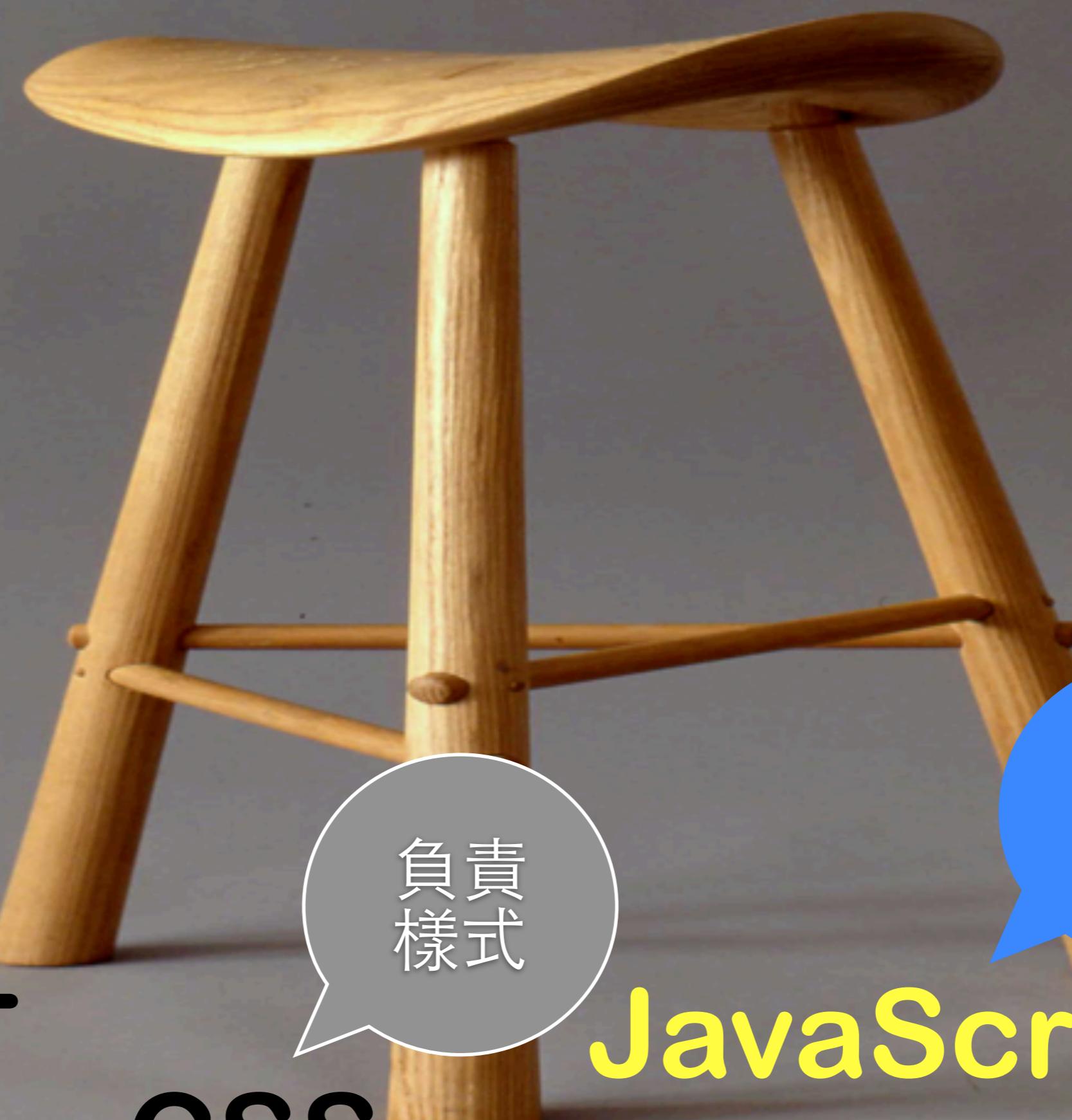


賦予生命 JavaScript

前端工程開發實務訓練 (3/8)

講師：蔣定宇 / josephj



負責
結構

HTML

負責
樣式

CSS

負責
行為

JavaScript

正確認知

以漸進式支援的方式來開發網站

你的 HTML 規劃了結構、再用了 CSS 紿予外觀

你的網站以可讓人觀看所有內容

JavaScript 是最後才實作的 Layer

主要目的在於增強網頁行為、讓使用者更好用

應避免一定要 JavaScript 才能使用的限制

正確認知

- 跟 Java 是完全不同的兩種語言。
- JavaScript 並不只存在 Browser 中
 - ❖ Command-Line 命令行
 - ❖ Server-side 後端伺服器語言
- Browser 裡多了 DOM 的 API、讓 JavaScript 能存取 HTML。

JavaScript 的使用方式

著重於瀏覽器上的開發

行間 JavaScript

直接在標籤內使用 `on<事件>` 屬性

```
<p on<事件>="JS 程式碼">...</p>
```

直接在 `a` 標籤的 `href` 屬性使用屬性

```
<a href="javascript:JS程式碼">...</a>
```

不推薦！將破壞 HTML 只有結構的原則！

嵌入式 JavaScript

直接將 JavaScript 寫在 HTML 內的作法

<script></script> 標籤

type 可省略

```
<script type="text/javascript">
```

JavaScript 程式碼

```
</script>
```

建議放置位置: </body> 之前

不推薦！將破壞 HTML 只有結構的原則！

只有入口網站需要（速度較快）

外部 JavaScript

引用外部 JavaScript 檔案（建議附檔名為 .js）

<script/> 標籤配合 src 屬性

type 可省略

```
<script type="text/javascript" src="JS 檔案路徑"></script>
```

與外部檔案連結

```
var name = "Awoo";
alert("Hello " + name);
```

※ 不需 script 標籤

推薦！維持 HTML 只有結構的原則！

練習用 HTML

http://f2e阶级.com/lab/js/_template.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>JavaScript</title>
</head>
<body>
<script>
// 在這邊寫 JavaScript 程式碼

</script>
</body>
</html>
```

雖然不推薦嵌入式，但這樣練習起來比較方便 :)

JavaScript 基礎

雖然不怎麼有趣、但對基礎是很重要的喔！

JavaScript 語法

我的第一個 JavaScript

Hello! <Name>

宣告 變數名稱 (大小寫有別)

變數值

字串需用引號

單雙引號皆可

```
var name = "Awoo";
```

```
alert("Hello! " + name);
```

字串合併、數值的加號都是用 +

JavaScript 語法

我的第一個 JavaScript

Hello! <Name>

```
var name = "Awoo";
alert("Hello! " + name);
```

練習看看

<http://f2eclass.com/lab/js/hello.html>

Variable 變數

顧名思義是變動的值

```
var name = "Awoo";  
name = name + " Chiang";  
alert(name); // "Awoo Chiang"
```

Variable 變數

未定義型別

<http://f2e阶级.com/lab/js/undefined.html>

```
alert(fruits);
```

Uncaught ReferenceError:
fruits is not defined

執行到未宣告的變數，
會產生錯誤並中止執行

```
var fruits;
```

```
alert(fruits); // undefined
```

有宣告就不會錯誤

<http://f2e阶级.com/lab/js/undefined-2.html>

Variable 變數

數字型別

```
var age = 18;
```

Variable 變數

型別間的轉換

型別是非常鬆散的

可以用 `typeof` 關鍵字觀看型別

```
var age = 18 + ""; // "18"  
age = age*2; // 36
```

Variable 變數

布林值 型別

true / false

```
var isBoy = true;
```

Variable 變數

陣列型別

```
var fruits = ["kiwifruit", "grape",  
"orange", "starfruit", "guava"];
```

```
alert(fruits[0]); // "kiwifruit"  
alert(fruits[4]); // "guava"  
alert(fruits[5]); // undefined
```

陣列的索引從 0 開始計算

Variable 變數

物件型別

```
var students = {  
    knock: {  
        gender: "male",  
        position: "manager"  
    },  
    allen: {  
        gender: "male",  
        position: "engineer"  
    }  
};  
alert(students.knock.position); // "manager"
```

目前網路上主要的資料交換型態 - JSON

Variable 變數

函式

```
var getStudentNames = function () {  
    // do something here...  
};  
不能省略  
↑  
要有分號!
```

大括號、可換行

Variable 變數

函式 - 參數、回傳值

參數：讓函式使用起來更彈性

```
var getStudentPosition = function (data) {  
    return data.knock.position;  
};
```

回傳值（不一定要有）



函式的使用：

```
var position = getStudentPosition(students);  
alert(position);
```

JavaScript 函式缺乏...

- 不提供 private (有替代作法)
- 多載 Overload - 無法依參數數量不同執行同名但不同的函式。
- 無法直接替參數指定預設值。

Closure

函式中有非區域變數也非參數的變數

```
var foo = 1;
```

```
var doSomething = function () {  
    alert(foo);  
    foo = foo + 1; ←————— foo 並不屬於此函式、是外界的變數  
};
```

```
doSomething();
```

```
alert(foo); // 2
```

```
doSomething();
```

```
alert(foo); // 3
```

不同時期執行 foo 值會有所不同
證明了：foo 是被保存變數而非變數值

Closure

更複雜一點...

```
var doOne = function () {  
    var x = 10;  
    return function (y) {  
        return x + y;  
    };  
};
```

x 並不屬於此函式、是外界的變數

```
var doOther = doOne();  
alert("doOther(10) : " + doOther(10));  
alert("doOther(20) : " + doOther(20));
```

↑
x 變數仍然被 Closure 保留著。

Scope

作用域

```
var foo = 1;
```

```
doSomething = function () { ←
```

```
    var foo = 2;
```

函式會形成一個作用域

```
    foo = foo + 1;
```

```
} ; ←
```

```
doSomething();  
alert(foo); // ?
```

全域變數

未透過 var 或在底層宣告的，會影響所有的 JS

```
var doSomething = function () {  
    foo = 1;  
};  
doSomething();
```

缺乏 var 告知，一旦 doSomething 執行、
foo 會變成全域變數。

```
<script type="text/javascript">  
  
var foo = 1;  
  
</script>
```

在最上層宣告，同樣是全域變數

注意：禁止使用全域變數、會非常混亂！

匿名函式

函式不一定都需要有名字的

```
foo.onclick = doSomething; ← 點選時執行某個具名函式
```

```
foo.onclick = function () { ← 點選時執行匿名函式  
    // doSomething;  
};
```

```
<script type="text/javascript">  
(function () {  
    var foo = 1;  
}());  
</script>
```

透過匿名函式產生作用域
解決全域變數的問題！

運算子 Operator

加減乘除

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic between variables and/or values.

Given that **y=5**, the table below explains the arithmetic operators:

Operator	Description	Example	Result of x	Result of y
+	Addition	<code>x=y+2</code>	7	5
-	Subtraction	<code>x=y-2</code>	3	5
*	Multiplication	<code>x=y*2</code>	10	5
/	Division	<code>x=y/2</code>	2.5	5
%	Modulus (division remainder)	<code>x=y%2</code>	1	5
++	Increment	<code>x=++y</code>	6	6
		<code>x=y++</code>	5	6
--	Decrement	<code>x=--y</code>	4	4
令人混淆、避免使用！		<code>x=y--</code>	5	4

http://www.w3schools.com/js/js_operators.asp

運算子 Operator

比較運算子

Given that **x=5**, the table below explains the comparison operators:

Operator	Description	Comparing	Returns
==	is equal to	<code>x==8</code>	<i>false</i>
		<code>x==5</code>	<i>true</i>
====	is exactly equal to (value and type)	<code>x===="5"</code>	<i>false</i>
		<code>x====5</code>	<i>true</i>
!=	is not equal	<code>x!=8</code>	<i>true</i>
!==	is not equal (neither value or type)	<code>x!==="5"</code>	<i>true</i>
		<code>x!==5</code>	<i>false</i>
>	is greater than	<code>x>8</code>	<i>false</i>
<	is less than	<code>x<8</code>	<i>true</i>
>=	is greater than or equal to	<code>x>=8</code>	<i>false</i>
<=	is less than or equal to	<code>x<=8</code>	<i>true</i>

http://www.w3schools.com/js/js_comparisons.asp

運算子 Operator

邏輯運算子

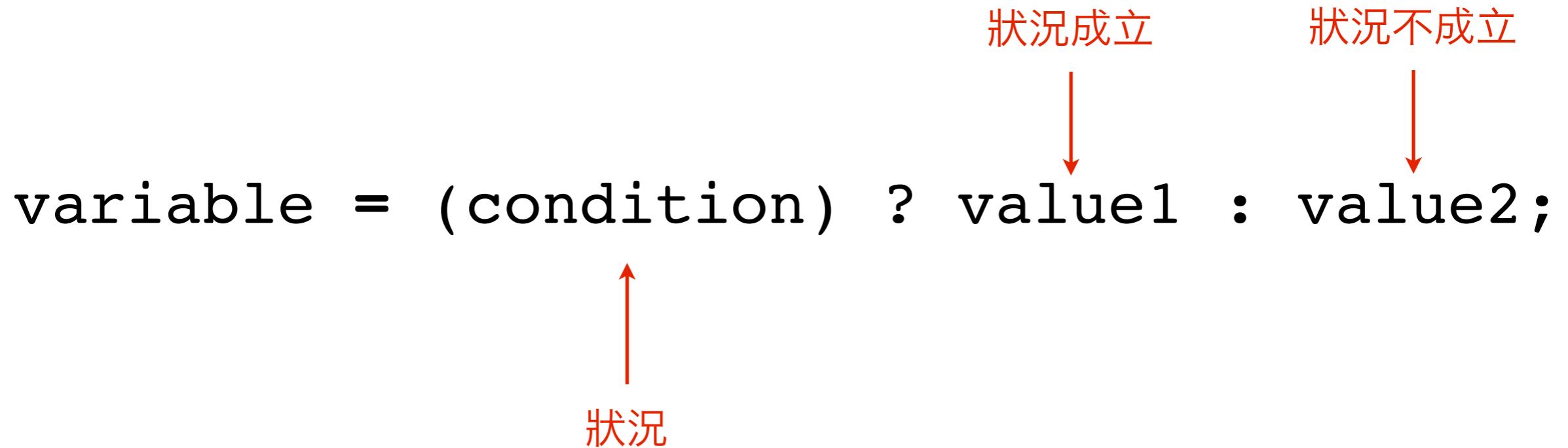
Given that **x=6** and **y=3**, the table below explains the logical operators:

Operator	Description	Example
&&	and	$(x < 10 \&\& y > 1)$ is true
	or	$(x==5 y==5)$ is false
!	not	$!(x==y)$ is true

http://www.w3schools.com/js/js_comparisons.asp

運算子 Operator

Conditional 運算子



http://www.w3schools.com/js/js_comparisons.asp

流程判斷

if... else...

```
if (狀況1) {  
}  
} else if (狀況2) {  
}  
} else if (狀況3){  
}  
} else { // 其他所有情況  
}
```

練習： if... else...

- 寫一個判斷式，hour 變數在：
 - ❖ 6 ~ 11 時：顯示 Good morning!
 - ❖ 12 ~ 17 時：顯示 Good afternoon!
 - ❖ 18 ~ 23 時：顯示 Good evening!
 - ❖ 24 ~ 5 時：顯示 Good night!

<http://class.josephj.com/lab/js/js-if-else.html>

流程判斷

switch

```
var weekday = 1;  
switch (weekday) {  
case 1:  
    alert("星期一");  
    break;  
case 2:  
    alert("星期二");  
    break;  
case 3:  
    alert("星期三");  
    break;  
case 4:  
    alert("星期四");  
    break;  
case 5:  
    alert("星期五");  
    break;  
case 6:  
    alert("星期六");  
    break;  
case 7:  
    alert("星期日");  
    break;  
default:  
    alert("什麼日子都不是！");  
}
```

迴圈

for loop

```
for (i = 開始值; i < 結束值; i++) {  
}
```

很多資料處理、重複性工作都需要用到迴圈

問題：從 1 加到 10 該怎麼做？

迴圈

foreach

主要針對物件類型的變數做處理

```
foreach (i in obj) {  
    alert(i);          // 此物件的第 n 個屬性的屬性名稱  
    alert(obj[i]);    // 此物件的第 n 個屬性的屬性值  
}
```

練習：列出所有學生的名字

<http://class.josephj.com/lab/js/js-foreach.html>

練習：函式

```
var getStudentPosition = function (data) {  
    return data.knock.position;  
};  
  
var students = {  
    knock: {  
        gender: "male",  
        position: "manager"  
    },  
    allen: {  
        gender: "male",  
        position: "engineer"  
    }  
};  
  
var position = getStudentPositoin(students);  
alert(position);
```

物件導向基礎

簡單介紹物件導向

物件導向基本觀念

- 物件是獨立的個體，有自己的屬性及方法（人）
 - ❖ 屬性是此個體的資料
例如人的「姓名」、「身高」、「體重」
 - ❖ 方法是此個體的動作
例如人「講話」、「走路」
 - ❖ 事件是此個體會碰到的狀況
通常在方法執行後，例如人走路後碰到了「跌倒」事件

關於物件

如果把一個人當成一個物件...



```
var kvzhuang = {
    "綽號": "小莊",
    "英文名": "Kevin",
    "性別": "男",
    "星座": "巨蟹",
    "心情指數": 10,
    "烹飪": function () {},
    "走路": function () {},
    "攀岩": function () {},
    "寫程式": function () {}
};
```

屬性

綽號：小莊

英文名：Kevin

性別：男

星座：巨蟹

方法

烹飪

走路

攀岩

寫程式

事件

烹飪燙到

烹飪完畢

攀岩完畢

程式有蟲

JavaScript 中、所有的型別都是物件！
都擁有自己的屬性跟方法！！

字串物件

任何字串變數都有以下屬性或函式

- **length** 此字串總長度
- **replace(from, to)** 把 from 改為 to
- **indexOf(str)** 從字串中找到 str 的位置
- **substr(x, y)** 以索引值 x 開始取 y 個字
- **substring(x, y)** 以索引值 x ~ y 取得部份內容
- **split()** 拆開成陣列

http://www.w3schools.com/jsref/jsref_obj_string.asp

陣列物件

任何陣列變數都有以下屬性或函式

- **length** 此陣列項目數量
- **push(item)** 將 item 放入此陣列中
- **join(str)** 將此陣列用 str 合併為字串
- **pop()**
- **splice()**

http://www.w3schools.com/jsref/jsref_obj_array.asp

日期物件

任何日期變數都有以下函式！

- `getHours()` 幾點 (0~23)
- `getDate()` 幾號 (1~31)
- `getMinutes()` 幾分 (0~59)
- `getTime()` 從 1970/1/1 0 時所經歷的毫秒數 (timestamp)

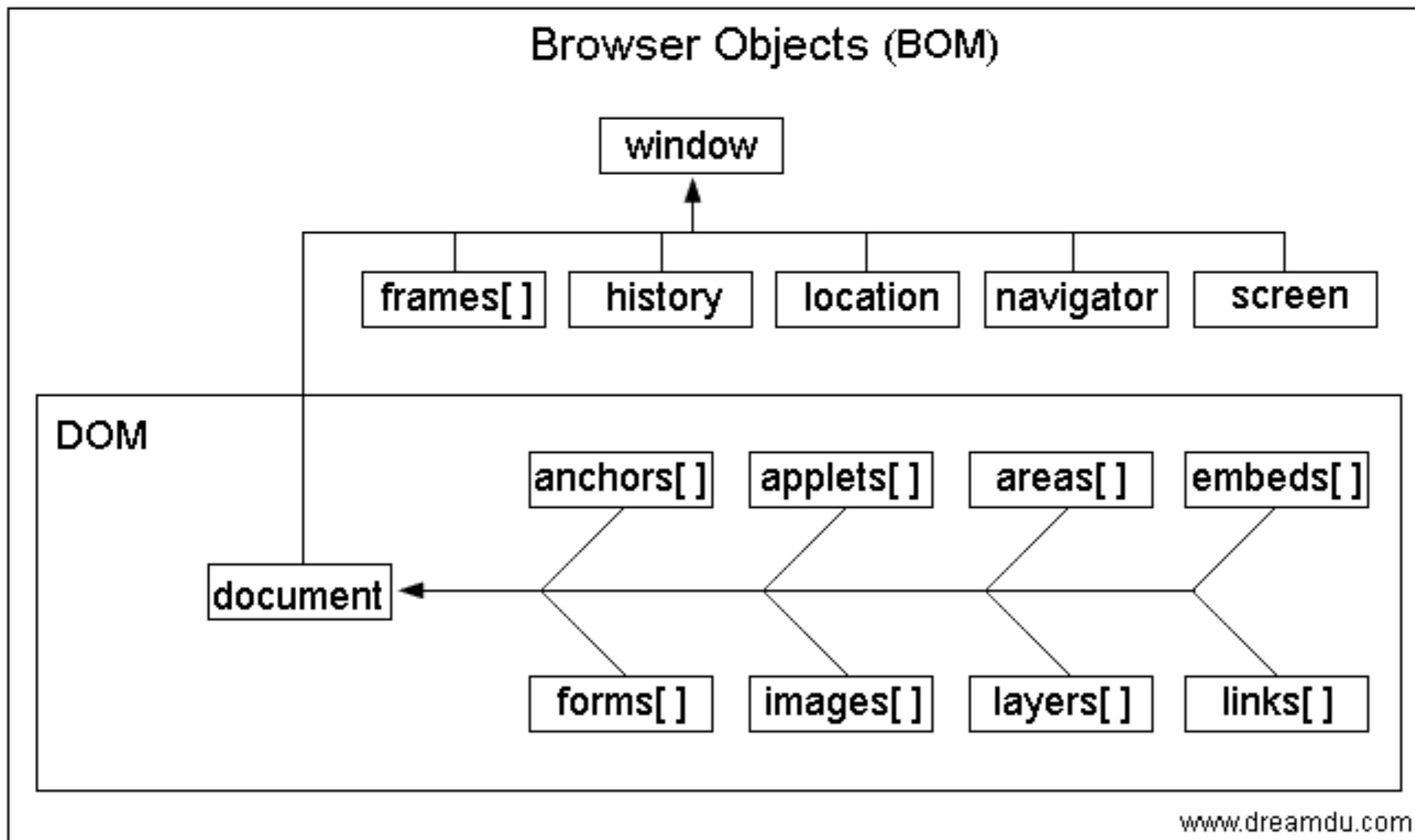
http://www.w3schools.com/jsref/jsref_obj_string.asp

BOM

瀏覽器物件模型

- Browser Object Model
- 由 Netscape 瀏覽器所制定
- 提供了 JS 對瀏覽器存取的 API 介面
- 核心物件為 window
- 不算是標準，但有時還是得用

BOM



現今較少使用 BOM 中 document 以下的物件

DOM

文件物件模型

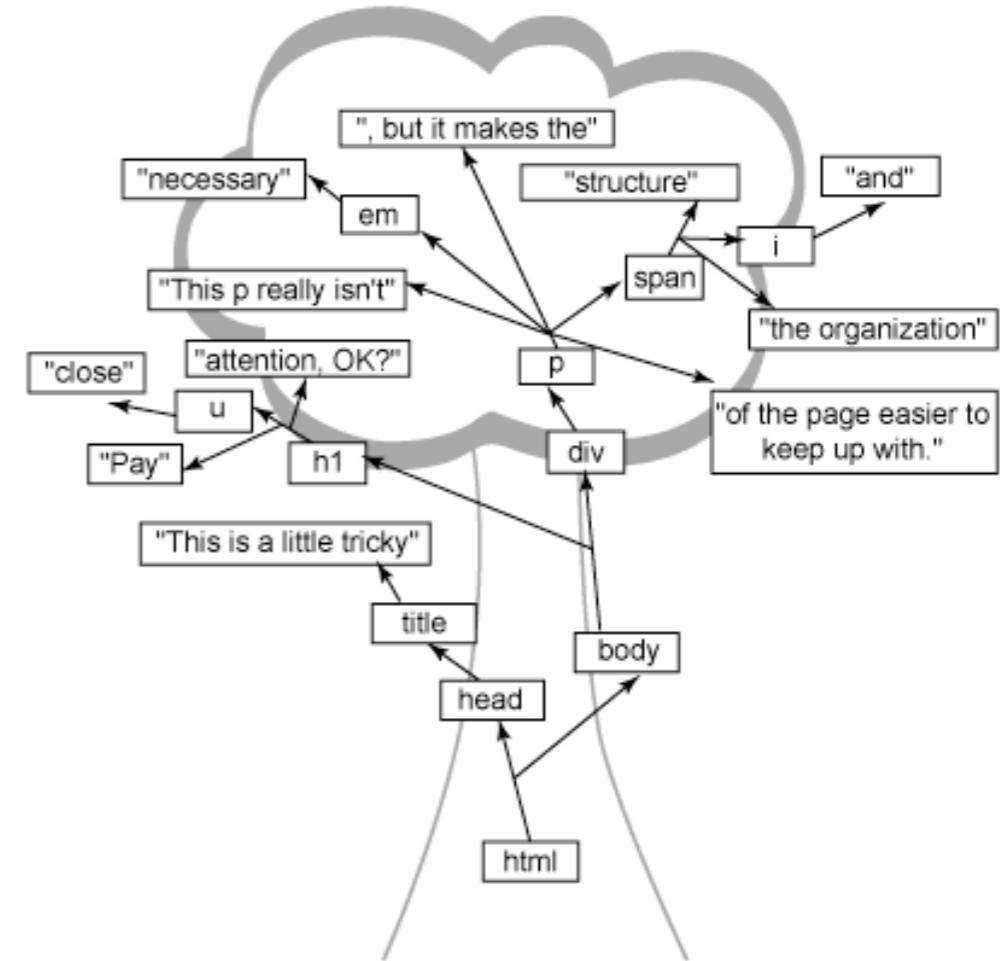
- Document Object Model
- 提供了 JS 對瀏覽器存取的 API 介面
- 核心物件為 document
- 由 W3C 持續維護的標準
- DOM 將整個 HTML 視為許多的節點、並提供許多 API 方法存取、新增、刪除、修改節點
- 我們只是透過 JavaScript 來操作 DOM API

DOM Tree

```

<!DOCTYPE html>
<html>
<head>
<title>Trees, trees, everywhere</title>
</head>
<body>
  <h1>Trees, trees, everywhere</h1>
  <p>Welcome to a <em>really</em> boring page.</p>
  <div>
    Come again soon.
    
  </div>
</body>
</html>

```



標籤、文字、屬性都是 DOM 的一部分

如何存取這些節點？

```
<div id="article">
    <h1>新店市好吃的餐廳</h1>

    <ul class="restaurant">
        <li>可愛村便當：台北縣新店市大豐街8號</li>
        <li>阿輝現炒黃牛肉：台北縣新店市中興路三段225號</li>
        <li>黑人牛肉麵：台北縣新店市中興路三段176號</li>
        <li>蘇杭：台北縣新店市民權路25號2F</li>
        <li>王家包子：台北縣新店市國校路25號之1</li>
        <li>辣。四川成都小炒：台北縣新店市吉安路2-1號</li>
        <li>山東餃子館：台北縣新店市中央路133巷19號1樓</li>
        <li>樂陶然健康小火鍋：台北縣新店市中央五街2號</li>
        <li>六六火鍋：台北縣新店市中央路89號</li>
        <li>龍門客棧：台北縣新店市中央路98號之1</li>
        <li>麵對麵：台北縣新店市中央路33號</li>
    </ul>
</div>
```

如何取得節點

- `document.getElementById()`
- `document.getElementsByTagName()`
- `document.getElementsByClassName()`
- BOM 所提供的方法（不建議）
- 很長一串？後面介紹 JavaScript 函式庫時都可以輕鬆解決！

<http://f2eclass.com/lab/js/dom-access.html>

常用的屬性或方法

- innerHTML 讀寫其內容
- nodeName 取得標籤名稱
- nodeValue 取得標籤的值
- style 設定此標籤的樣式
- 遊走節點: parentNode, childNodes, nextSibling, previousSibling
- 處理節點: appendChild, removeChild, cloneNode

<http://f2e阶级.com/lab/js/dom-attribute.html>
<http://f2e阶级.com/lab/js/dom-method.html>

節點的操作

- 新增一個節點到網頁 (DOM) 中。
 - ❖ 建立物件參考(還沒實際放到 DOM 中)
 - ◆ var el = document.createElement("標籤名稱");
 - ❖ 設定此節點的屬性與內容，例如：
 - ◆ el.innerHTML = "blahblahblah"
 - ❖ 針對要掛載的母節點、使用 appendChild / insertBefore 屬性
 - ◆ var list = document.getElementById("photos");
list.appendChild(el);
- 刪除一個節點
 - ❖ 先取得此節點參考，再以母節點做刪除。
 - ❖ el.parentNode.removeChild(el); // 有夠囉嗦的 XD
- 複製使用 cloneNode、移動使用 appendChild

練習：建立節點

<http://f2e阶级.com/lab/js/dom-appendchild.html>

將陣列轉換為 li 列表

```
var items = [
    "可愛村便當：台北縣新店市大豐街8號",
    "阿輝現炒黃牛肉：台北縣新店市中興路三段225號",
    "黑人牛肉麵：台北縣新店市中興路三段176號",
    "蘇杭：台北縣新店市民權路25號2F",
    "王家包子：台北縣新店市國校路25號之1",
    "辣。四川成都小炒：台北縣新店市吉安路2-1號",
    "山東餃子館：台北縣新店市中央路133巷19號1樓",
    "樂陶然健康小火鍋：台北縣新店市中央五街2號",
    "六六火鍋：台北縣新店市中央路89號",
    "龍門客棧：台北縣新店市中央路98號之1",
    "麵對麵：台北縣新店市中央路33號"
];
for (var i = 0, j = items.length; i < j; i++) {
    items[i]; // 項目 i 的值
}
```

練習前面學到的迴圈、DOM 的取得與操作

DOM Event

- JavaScript 跟其他語言最大的不同：
「事件驅動」(Event-driven)
- 當使用者做了什麼事情時，就去執行某個函式
- DOM 提供了多樣化的 Event:
滑鼠事件、鍵盤事件、表單事件、手機事件...

事件列表

- **click** 滑鼠點擊
- **dblclick** 滑鼠雙點擊
- **mousemove** 滑鼠移動
- **mouseover** 滑鼠移入
- **mouseout** 滑鼠移出
- **mouseup** 滑鼠點選方開
- **keydown** 按鍵壓到底
- **keypress** 按鍵開始壓
- **keyup** 按鍵彈起來
- **blur** 焦點移除
- **change** 內容改變
- **focus** 焦點
- **reset** 表單重設
- **select** 選取文字
- **submit** 表單送出
- **abort** 中止傳送
- **error** 錯誤發生
- **load** 讀取完畢
- **resize** 文件改變大小
- **scroll** 滑鼠滾輪事件
- **unload** 文件卸除
- **beforeunload**

DOM Event 繩法

- 直接在標籤內使用 `on<事件>` 屬性
 - ❖ `<p on<事件>="JS 程式碼">...</p>`
 - ❖ 不建議。會破壞 HTML 只負責結構的原則。
- 物件參考`.on("事件名稱", 指定函式);`
 - ❖ 缺點：一個物件只能對一種事件繩一次，後面指定的會覆寫。
- 物件參考`.addEventListener("事件名稱", 指定函式);`
 - ❖ 優點：一個物件可以對多個事件繩多次，後面指定的優先執行。
 - ❖ 缺點：IE 舊版本的瀏覽器不支援。(IE 是用 `attachEvent`)

練習：結合事件

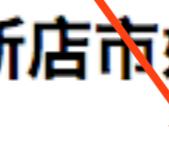
<http://f2eclass.com/lab/js/dom-event.html>

新增、刪除、修改節點的動作都由使用者操作

列出餐廳 單雙行區隔 移除列表

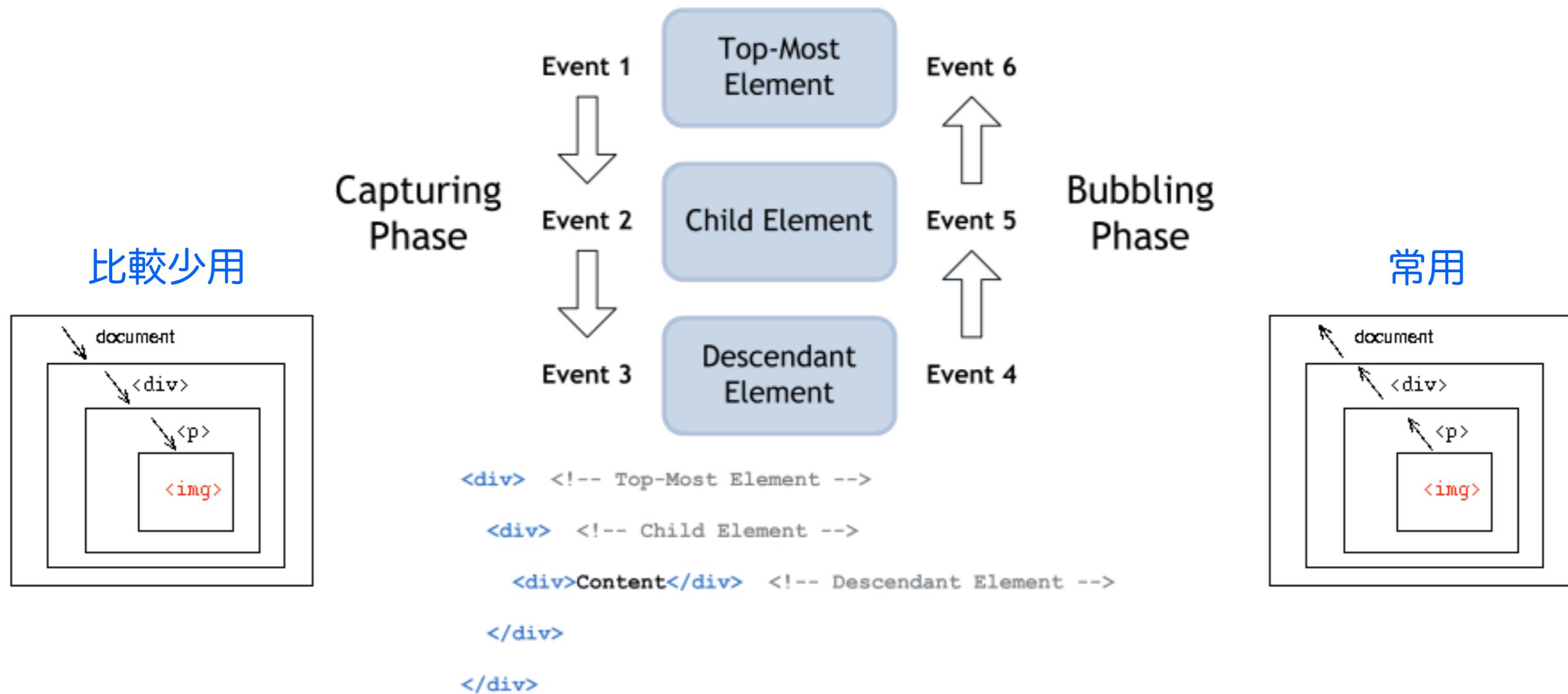
新店市好吃的餐廳

- 可愛村便當：台北縣新店市大豐街8號
- 阿輝現炒黃牛肉：台北縣新店市中興路三段225號
- 黑人牛肉麵：台北縣新店市中興路三段176號
- 蘇杭：台北縣新店市民權路25號2F
- 王家包子：台北縣新店市國校路25號之1
- 辣。四川成都小炒：台北縣新店市吉安路2-1號
- 山東餃子館：台北縣新店市中央路133巷19號1樓
- 樂陶然健康小火鍋：台北縣新店市中央五街2號
- 六六火鍋：台北縣新店市中央路89號
- 龍門客棧：台北縣新店市中央路98號之1
- 麵對麵：台北縣新店市中央路33號



動作快的同學請幫我結合 Bootstrap!

事件處理機制



有時要綁的物件可能有幾百個，
可以用氣泡的概念綁最外層的即可 (委任)

課程回顧

- 漸進式支援：替網頁加上 JavaScript 後，提供了行為、方便使用者操作。(但非必須)
- JavaScript 是程式語言，可存在於各種平台。
- 處理瀏覽器的節點與事件由 DOM API 所提供的、JavaScript 只是媒介。
- DOM API 在每個瀏覽器的實作不同，寫起來也麻煩，因此造就了像 YUI 與 jQuery 這樣 JavaScript Library 的誕生。
- 本課程主要在建立基礎與正確認知，未來的 DOM 操作會使用 Library 來取代。

Q & A