



Chapter 05

JavaScript內建物件

中原大學 資訊管理學系

賴錦慧 老師

chlai@cycu.edu.tw

大綱

- JavaScript的內建物件
- JavaScript的String物件
- JavaScript的Array物件
- JavaScript的Date物件
- JavaScript的Math物件
- JavaScript的Error物件
- 物件的共用屬性和方法

JavaScript的內建物件



- JavaScript內建物件的種類
- JavaScript的內建物件

JavaScript內建物件的種類



□ 隱性物件 (Implicit objects)

- JavaScript的各種資料型態變數，在宣告和指定值後就是一個物件

```
var str="JavaScript網頁程式設計";
```

□ 顯性物件 (Explicit objects)

- JavaScript物件使用new運算子建立物件

```
var str= new String("JavaScript網頁程式設計");
```

JavaScript的內建物件(1/3)



- JavaScript提供十一種內建物件，常見的其他物件如下：

Boolean物件

- Boolean物件是一種資料型態
- 提供建構函數可以建立布林資料型態的物件
`var objBoolean = new Boolean();`

Function物件

- JavaScript函數就是一個Function物件
`var mod = new Function("x", "y", "return(x%y)");`

```
function mod(x, y) {  
    return (x%y);  
}
```

JavaScript的內建物件(2/3)



Global物件

- Global物件不能使用new運算子建立，在腳本語言引擎初始後就會自動建立此物件。
- 直接使用屬性名稱: Infinity, NaN

Number物件

- 建立數值資料型態的變數
`objnum = new Number(value);`
- 使用Number物件的目的是為了使用toString()方法將數值轉換成字串。

JavaScript的內建物件(3/3)



Object物件

□ Object物件可以建立JavaScript支援的物件

`objobject = new Object(value);`

- 參數value是String → 字串物件
- 參數value是Boolean → Boolean物件

RegExp物件

□ RegExp物件是JavaScript「正規運算式」
(Regular Expression) 物件。

JavaScript的String物件



- 建立String物件
- 字串長度與大小寫
- 取得字串的指定字元
- 子字串的搜尋
- 子字串的處理

建立String物件

□String物件

- 處理字串變數的資料
- 進行子字串操作

```
var objstr1="JavaScript";  
var objstr2= new String("網頁程式設計");
```

JavaScript String Reference

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

HTML標籤格式編排



名稱	說明
anchor()	傳回<a>string標籤字串
big()	傳回<big>string</big>標籤字串
blink()	傳回<blink>string</blink>標籤字串(I.E不會閃)
bold()	傳回string標籤字串
fixed()	傳回<tt>string</tt>標籤字串
fontcolor(color)	傳回string標籤字串
fontsize(size)	傳回string標籤字串
italics()	傳回<i>string</i>標籤字串
link(url)	傳回string標籤字串
small()	傳回<small>string</small>標籤字串
strike()	傳回<strike>string</strike>標籤字串
sub()	傳回_{string}標籤字串
sup()	傳回^{string}標籤字串

Example

- 把 HTML 標記加入字串，方便輸出 HTML 格式文字

```
<script>  
var text = "Affected  
Text";  
alert(text.big());  
</script>
```

<big> Affected Text</big>

確定

```

<script>
var winvar
var text = "Affected Text"

function dw(line) {
winvar.document.writeln(line + "<br>");
}

function outputwin1() {
winvar = window.open();
document.write("text.big(): " + text.big());
document.write("text.blink(): " + text.blink());
document.write("text.bold(): " + text.bold());
document.write("text.italics(): " + text.italics());
document.write("text.fixed(): " + text.fixed());
document.write("text.fontcolor('yellow'): " + text.fontcolor('yellow'));
document.write("text.fontcolor('#FFCC99'): " + text.fontcolor('#FFCC99'));
document.write("text.fontcolor(16581375): " + text.fontcolor(16581375));
for (i=1 ; i<=7 ; i++) {
    document.write("When fontsize = " + i + ", " + text.fontsize(i));
}
document.write("text.small(): " + text.small());
document.write("text.strike(): " + text.strike());
document.write("text.sub(): " + text.sub());
document.write("text.sup(): " + text.sup());
}
</script>

```

```

text.big(): Affected Text
text.blink(): Affected Text
text.bold(): Affected Text
text.italics(): Affected Text
text.fixed(): Affected Text
text.fontcolor('yellow'): Affected Text
text.fontcolor('#FFCC99'): Affected Text
text.fontcolor(16581375): Affected Text
When fontsize = 1, Affected Text
When fontsize = 2, Affected Text
When fontsize = 3, Affected Text
When fontsize = 4, Affected Text
When fontsize = 5, Affected Text
When fontsize = 6, Affected Text
When fontsize = 7, Affected Text
text.small(): Affected Text
text.strike(): Affected Text
text.sub(): Affected Text
text.sup(): Affected Text

```

字串長度與大小寫

□ **String**物件提供方法和屬性可以取得字串長度和英文字串的大小寫轉換

■ **length**屬性：取得字串的長度。

□ **String**物件方法

■ **toLowerCase()**方法：將字串的英文字母都轉換成小寫字母。

■ **toUpperCase()**方法：將字串的英文字母都轉換成大寫字母。

取得字串的指定字元

□ 取得字串指定位置的字元

- `charAt(index)` 方法：取得參數 `index` 位置的字元，索引值是從 0 開始

- `charCodeAt(index)` 方法：取得參數 `index` 位置的 Unicode 統一字碼

- 注意：參數 `index` 都是以 0 開始，第 1 個字元為 0，第 2 個字元為 1，依序類推。

子字串的搜尋

□ String物件提供功能強大的子字串搜尋方法

■ indexOf(string, index)方法

- 傳入參數是搜尋字串，index為開始搜尋的索引位置
- 傳回第1次搜尋到字串的索引位置，沒有找到傳回-1

■ lastIndexOf(string)方法

- 如同indexOf()方法，不過是從尾搜尋到頭的反向搜尋。

■ match(string)方法

- 如同indexOf()和lastIndexOf()
- 若有找到,傳回的為找到的字串;沒有找到則傳回null

■ search(string)方法

- 與indexOf()的功能相似

子字串的處理

- **String**物件提供方法可以取代、分割和取出字串中所需的子字串
 - **replace(string1, string2)**方法：將找到的**string1**子字串取代成**string2**。
 - **split(string)**方法：傳回**Array**物件，使用參數**string**作為分割字串，可以將字串轉換成**Array**物件。
 - **substr(index, length)**方法：從**index**開始取出**length**個字元。
 - **substring(index1, index2)**方法：取出**index1**到**index2**之間的字串。
 - **concat(string)**方法：將**string**字串新增到**String**物件的字串後。

練習1

□ 請設定下列2個字串變數, 並完成各題之計算

`str1="JavaScript"`

`str2="HTML5,JavaScript,jQuery網頁程式設計"`

- 1) 分別計算2個字串的長度
- 2) 在`str2`中搜尋字串"Query "
- 3) 將2個字串中的" JavaScript "取代為" JS "
- 4) 針對`str2`, 自第9個字元開始, 取出6個字元



JavaScript的Array物件

- JavaScript的一維陣列
- Array物件的屬性和方法
- JavaScript的多維陣列

JavaScript的一維陣列-建立(1/2)



- JavaScript陣列和物件的分野並不明顯，陣列擁有陣列元素如同物件擁有屬性，JavaScript陣列事實上就是一種特殊物件。
- JavaScript陣列的索引值是從0開始，JavaScript宣告陣列的方法就是建立Array物件
`var username = new Array(5);`
- 我們可以使用索引值來指定陣列的元素值，如下所示：
`username[0] = "Joe";`
`username[1] = "Jane";`
`...`
`username[4] = "Merry";`

JavaScript的一維陣列-建立(2/2)



- 在建立Array物件時，直接在參數指定陣列元素值

```
var tips = new Array(100,200,500);
```

- username[]陣列是一個字串陣列；tips[]陣列是數值陣列。

JavaScript的一維陣列-走訪



- 使用for迴圈走訪和顯示陣列元素

```
for (var i = 0; i < 5; i++) {  
    document.write(username[i] + "<br/>");  
}
```

- 程式碼使用陣列索引值取得每一個陣列元素的值，迴圈的結束條件是使用length屬性取得陣列尺寸。

Array物件的屬性和方法

- Array物件提供屬性和方法可以取得陣列尺寸、排序陣列元素、合併陣列和反轉陣列元素。
- Array物件的屬性
 - **length**屬性：取得陣列的元素個數，即陣列尺寸。
- Array物件的相關方法：
 - **join()**方法：將陣列的元素使用字串方式顯示，每個陣列元素是使用「,」符號分隔。
 - **reverse()**方法：將陣列元素反轉，本來是陣列的最後一個元素成為第一個元素。
 - **sort()**方法：將陣列所有元素進行排序。
 - **concat(array)**方法：將參數的陣列合併到目前的陣列中。

JavaScript的多維陣列-建立



□ 建立多維陣列

```
var users = new Array(5);  
for(var i = 0; i < 5; i++)  
    users[i] = new Array(2);
```

□ 指定二維陣列的元素值，如下所示：

```
users[0][0] = "Joe";  
users[0][1] = "1234";  
users[1][0] = "Jane";  
users[1][1] = "5678";  
  
...  
users[4][0] = "Merry";  
users[4][1] = "5678";
```

□ 可以將Array物件擴充成多維陣列。

練習2

□ 請建立一個二維矩陣，每個陣列元素儲存 $i*j=n$ ，最後再列印九九乘法表的結果(下三角)

九九乘法表

1*1 = 1	1*2 = 2	1*3 = 3	1*4 = 4	1*5 = 5	1*6 = 6	1*7 = 7	1*8 = 8	1*9 = 9
2*1 = 2	2*2 = 4	2*3 = 6	2*4 = 8	2*5 = 10	2*6 = 12	2*7 = 14	2*8 = 16	2*9 = 18
3*1 = 3	3*2 = 6	3*3 = 9	3*4 = 12	3*5 = 15	3*6 = 18	3*7 = 21	3*8 = 24	3*9 = 27
4*1 = 4	4*2 = 8	4*3 = 12	4*4 = 16	4*5 = 20	4*6 = 24	4*7 = 28	4*8 = 32	4*9 = 36
5*1 = 5	5*2 = 10	5*3 = 15	5*4 = 20	5*5 = 25	5*6 = 30	5*7 = 35	5*8 = 40	5*9 = 45
6*1 = 6	6*2 = 12	6*3 = 18	6*4 = 24	6*5 = 30	6*6 = 36	6*7 = 42	6*8 = 48	6*9 = 54
7*1 = 7	7*2 = 14	7*3 = 21	7*4 = 28	7*5 = 35	7*6 = 42	7*7 = 49	7*8 = 56	7*9 = 63
8*1 = 8	8*2 = 16	8*3 = 24	8*4 = 32	8*5 = 40	8*6 = 48	8*7 = 56	8*8 = 64	8*9 = 72
9*1 = 9	9*2 = 18	9*3 = 27	9*4 = 36	9*5 = 45	9*6 = 54	9*7 = 63	9*8 = 72	9*9 = 81

九九乘法表

1*1 = 1								
2*1 = 2	2*2 = 4							
3*1 = 3	3*2 = 6	3*3 = 9						
4*1 = 4	4*2 = 8	4*3 = 12	4*4 = 16					
5*1 = 5	5*2 = 10	5*3 = 15	5*4 = 20	5*5 = 25				
6*1 = 6	6*2 = 12	6*3 = 18	6*4 = 24	6*5 = 30	6*6 = 36			
7*1 = 7	7*2 = 14	7*3 = 21	7*4 = 28	7*5 = 35	7*6 = 42	7*7 = 49		
8*1 = 8	8*2 = 16	8*3 = 24	8*4 = 32	8*5 = 40	8*6 = 48	8*7 = 56	8*8 = 64	
9*1 = 9	9*2 = 18	9*3 = 27	9*4 = 36	9*5 = 45	9*6 = 54	9*7 = 63	9*8 = 72	9*9 = 81

JavaScript的Date物件



- 取得日期和時間
- 設定日期和時間
- 日期和時間的轉換
- 取得系統的時間

取得日期和時間-建立Date物件



□ Date物件在使用new運算子建立物件後，
就可以取得系統的時間和日期

```
var dttoday = new Date();
```



取得日期和時間-相關方法

方法	說明
getDate()	傳回日期值1~31
getDay()	傳回星期值0~6，也就是星期日到星期六
getMonth()	傳回月份值0~11，也就是一到十二月
getFullYear()	傳回完整年份，例如：2011
getYear()	傳回年份，如果在1900~1999年之間，傳回後兩碼，例如：1999年傳回99，否則傳回完整年份
getHours()	傳回小時0~23
getMinutes()	傳回分鐘0~59
getSeconds()	傳回秒數0~59
getMilliseconds()	傳回千分之一秒為單位的秒數，0~999
getTime()	傳回自1/1/1970年開始的秒數，以千分之一秒（毫秒）為單位

設定日期和時間



□ **Date**物件提供方法存取日期與時間，如下表：

方法	說明
setDate()	設定Date物件的日期1~31
setMonth()	設定Date物件的月份0~11
setFullYear()	設定Date物件的完整年份
setYear()	設定Date物件的年份，在1990~1999間只需使用後兩位，例如： 1999使用99，否則需要使用完整年份
setHours()	設定Date物件的小時0~23
setMinutes()	設定Date物件的分鐘0~59
setSeconds()	設定Date物件的秒數0~59
setMilliseconds()	設定Date物件的秒數，以千分之一秒為單位，0~999
setTime()	設定Date物件的時間，自1/1/1970年開始，以千分之一秒為單位



日期和時間的轉換

- **Date**物件提供日期和時間轉換方法，可以取得時間差、轉換成千分之一秒數或輸出成字串等轉換操作
- **GMT**為格林威治標準時間
 - **getTimezoneOffset()**方法：傳回本地時間和**GMT**的時間差，以分為單位。
 - **toGMTString()**方法：傳回轉換成**GMT**時間的字串。
 - **toLocaleString()**方法：傳回將**GMT**轉換成本地時間的字串。
 - **parse(Date)**方法：傳回參數**Date**物件從1/1/1970到本地時間的毫秒數，以千分之一秒為單位。
 - **UTC(Date)**方法：傳回參數**Date**物件從1/1/1970到**GMT**時間的毫秒數，以千分之一秒為單位。

取得系統的時間

- JavaScript的Date物件可以取得系統時間
 - 只需定時執行JavaScript函數，就可以使用Date物件建立網頁小時鐘。
 - JavaScript小時鐘需要使用Window物件的計時器方法setTimeout()，方法參數可以設定在間隔時間後執行指定函數或網頁，對應的clearTimeout()方法可以清除計時器。

練習3

□ 電子小時鐘

請先建立一個空白網頁，在網頁上利用**Date**物件來建立電子小時鐘。所顯示的時間會根據系統時間動態更新。

- 先取得目前的系統時間
- 將所取得的時間以數字圖片取代，再顯示於網頁上。
- 設定小時鐘的**CSS**

電子小時鐘

1 1 : 3 2 : 2 4

JavaScript的Math物件



- Math物件的屬性
- Math物件的亂數、最大和最小值
- Math物件的數學方法

Math物件的屬性



屬性	說明
E	自然數 $e=2.718281828459045$
LN2	$\ln 2=0.6931471805599453$
LN10	$\ln 10=2.302585092994046$
LOG2E	$\log_2 e=1.4426950408889633$
LOG10E	$\log e=0.4342944819032518$
PI	圓周率 $\pi=3.141592653589793$
SQRT1_2	根號 $\sqrt{1/2}=0.7071067811865476$
SQRT2	根號 $\sqrt{2}=1.4142135623730951$

Math物件的亂數、最大和最小值



□ Math物件提供建立亂數、最大值和最小值的方法

- `max(value1,value2)`方法：傳回2個參數中的最大值。
- `min(value1,value2)`方法：傳回2個參數中的最小值。
- `random()`方法：傳回亂數值。
- `round(value)`方法：將參數值四捨五入後傳回。

Math物件的數學方法



方法	說明
abs()	傳回絕對值
acos()	反餘弦函數
asin()	反正弦函數
atan()	反正切函數
ceil()	傳回大於或等於參數的最小整數
cos()	餘弦函數
exp()	自然數的指數 e^x
floor()	傳回大於或等於參數的最大整數
log()	自然對數
pow()	次方
sin()	正弦函數
sqrt()	傳回參數的平方根
tan()	正切函數

Example

```
<script>
document.write("最大值max(34, 78): " + Math.max(34,78) + "<br/>");
document.write("最小值min(34, 78): " + Math.min(34,78) + "<br/>");
document.write("四捨五入round(34.567):" + Math.round(34.567) + "<br/>");
document.write("四捨五入round(34.567):" + Math.round(34.467) + "<br/>");
document.write("亂數random(): " + Math.random() + "<br/>");

// 0-10的亂數
var no = Math.round(Math.random()*10);
document.write("0-10亂數: " + no + "<br/>");

// 0-100的亂數
no = Math.round(Math.random()*100);
document.write("0-100亂數: " + no + "<br/>");
</script>
```

Math物件的亂數、最大和最小

```
最大值max(34, 78): 78
最小值min(34, 78): 34
四捨五入round(34.567):35
四捨五入round(34.567):34
亂數random(): 0.32877598493359983
0-10亂數: 9
0-100亂數: 47
```

JavaScript的Error物件



- JavaScript的例外處理
- JavaScript多層的例外處理架構

JavaScript的例外處理-Error物件



- **Error**物件儲存JavaScript執行時產生的錯誤資訊，當JavaScript執行階段的錯誤產生後，**Error**物件會自動建立
 - **number**屬性：錯誤碼，這是一個32-bit值，其中後16-bit才是真正的錯誤碼。
 - **message**屬性：錯誤訊息字串。
 - **description**屬性：如同**message**屬性，這也是錯誤說明的字串。

JavaScript的例外處理-敘述

- JavaScript例外處理程式敘述是：`try/catch/finally`，可以處理JavaScript執行階段的錯誤

```
try {
```

```
  ...
```

```
}
```

```
catch(e) {
```

```
  // 例外處理
```

```
  ...
```

```
}
```

```
finally {
```

```
  ...
```

```
}
```

JavaScript需要例外處理的
程式碼

傳入的參數e是Error物件

- 可以取得Error物件屬性的錯誤資訊
- 建立例外處理的程式碼。

選擇性的程式區塊
不論例外是否產生，都會執行此
區塊的程式碼。

JavaScript多層的例外處理架構



```
try {  
    ...  
    try {  
        throw 運算式  
    }  
    catch(e) {    // 第二層的例外處理  
        ...  
        throw e; // 丟到外層的例外處理  
    }  
} catch(e) { // 第一層的例外處理  
    ...  
}  
finally {  
    ...  
}
```


Example

```
<script>
var x = 10;
try {
  x = y; // 測試的錯誤程式碼
}
catch(e) {
  // 例外處理的程式碼
  document.write("錯誤碼: " + (e.number & 0xFFFF) + "<br/>");
  document.write("錯誤說明(message): " + e.message + "<br/>");
  document.write("錯誤說明(description): " + e.description + "<br/>");
}
finally {
  // 顯示測試值
  document.write("<hr/>測試值x = " + x + "<br/>");
}
</script>
```

JavaScript的例外處理

錯誤碼: 0
錯誤說明(message): y is not defined
錯誤說明(description): undefined

測試值x = 10



物件的共用屬性和方法

- JavaScript物件的共用屬性
- JavaScript物件的共用方法

JavaScript物件的共用屬性

- JavaScript物件的constructor屬性是共用屬性
 - constructor屬性可以取得建立物件使用的建構函數名稱
 - JavaScript內建物件除了Global和Math物件外都支援此屬性。
- 在使用new運算子建立test物件後，就可以使用if條件檢查物件的建構函數：

```
if (test.constructor == String) {  
    ...  
}
```

JavaScript物件的共用方法

- JavaScript物件常用的共用方法有toString()和valueOf()，這兩個方法可以顯示物件的內容。

toString()方法

- toString()方法：傳回物件的內容，傳回值為字串
`object.toString();`

valueOf()方法

- valueOf()方法：傳回物件值，不過Math和Error物件不支援valueOf()方法
`object.valueOf();`

toString()方法輸出的物件內容



物件	行為
Array	Array 的元素會轉換為字串。產生的字串是串連的，並以逗號分隔。
Boolean	如果布林值為 true，則會傳回 "true"。否則會傳回 "false"。
Date	傳回日期的文字表示。
錯誤	傳回包含相關錯誤訊息的字串。
函式	傳回以下格式的字串，其中 <i>functionname</i> 是呼叫其 toString 方法的函式名稱： <div><pre>function functionname() { [native code] }</pre></div>
Number	傳回數字的文字表示。
String	傳回 String 物件的值。
預設值	傳回 "[object objectname]"，其中 <i>objectname</i> 是物件型別的名稱。

valueOf()方法輸出的物件內容



物件	傳回值
Array	傳回陣列執行個體。
Boolean	布林值。
Date	從 UTC 1970 年 1 月 1 日午夜起算，儲存在物件中的時間值 (以毫秒為單位)
函式	函式本身。
Number	數值。
物件	物件本身。這是預設值。
String	字串值。