JavaScript
jQuery

黃語昕

silvia.huang@gmail.com

課程所需知識

- HTML
- CSS

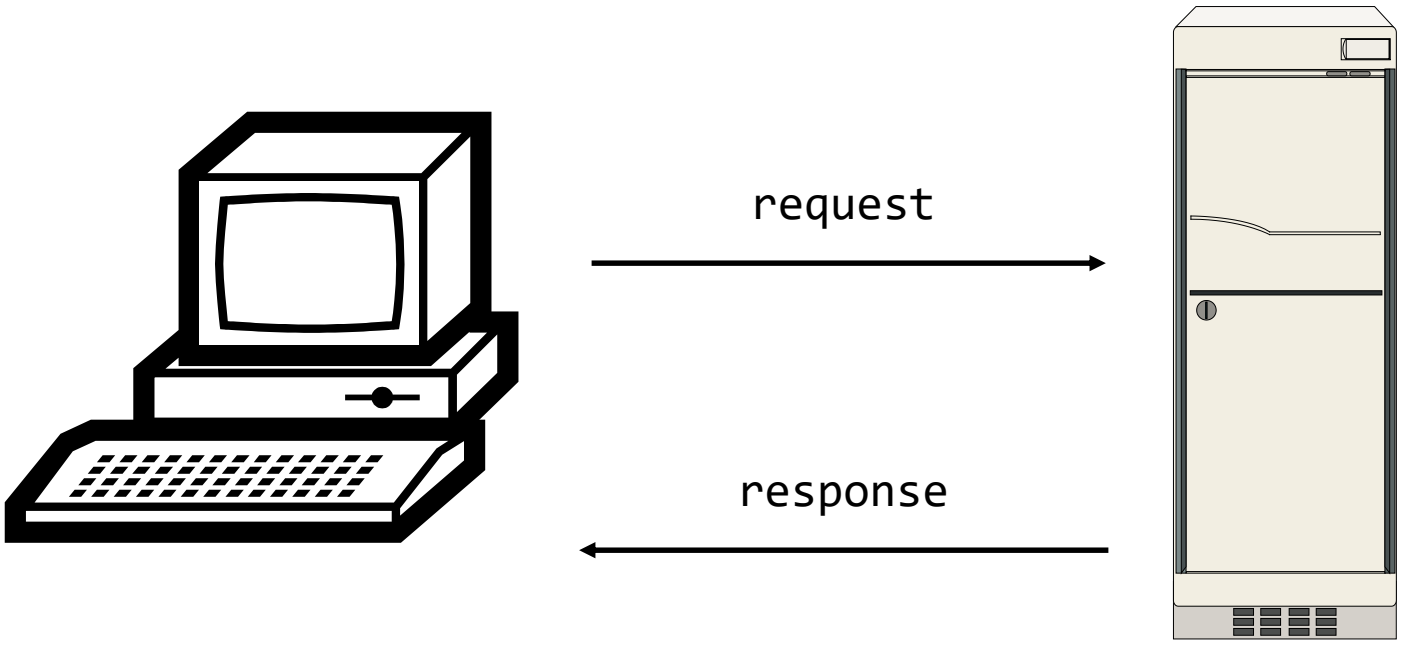
課程使用環境

- 作業系統
- VScode
- Google Chrome

環境介紹

1. Internet環境

   提供服務--E-mail、FTP、World Wide Web(WWW) …



2. 教室環境

(1)軟體需求

△瀏覽器(Browser) ：

- Internet Explorer
- Mozilla Firefox
- Google Chrome
- Opera
- Safari

△編寫工具(Text Editor)：

- Visual Studio Code
- Sublime Text
- Atom

(2)JavaScript console

(3)Web Server：

Windows：安裝Apache或IIS

Mac或Linux已有內建Apache

3. URL(Uniform Resources Locator)

△在Internet上尋找資源的一種通用的方式

△https://www.google.com/temp.html

包括了

- How："http:// " or “https:// " …
- Where：網址(ex. www.google.com)
- What：網頁or其他資源(ex. temp.html)

4. W3C 與 ECMA

5. ECMA 262

http://www.w3schools.com

JavaScript

What is JavaScript ?

- JavaScript是一個能在Browser(client-side)執行的應用程式，目的是為了彌補HTML互動性不足的問題
- 可內嵌於網頁內，JavaScript可以是HTML的一部分
- JavaScript是 client-side language，由Browser來解譯和執行，不需要網路也可以執行。
- 可在支援JavaScript的browser上執行。
- JavaScript易學難除錯
  - 以物件為基礎(Object-based)的語言
- HTML+CSS+JavaScript 線上編輯工具：
  https://codepen.io/pen
  https://jsbin.com/
  https://jsfiddle.net/

JavaScript 的優勢

- 互動式網頁
- 在client 端(Browser)執行即可
- 使用者資料驗證，減少server 端的負擔，減少網路壅塞
- 何時使用?
  - 增加網頁的可看性與互動性
  - 動態的資料呈現
  - 偵測user在網頁上的行為
  - 要求user輸入資料並驗證 (ex.加入會員)
  - 記錄使用者的選擇 (ex.購物車)

- **HTML5的Javascript APIs**

撰寫格式

- 放在HTML的<body>中，將JS當HTML的element來使用

```
<body>
    ...
</body>
```

- 放在HTML的<body>中，透過事件驅動來執行JS

```
<body>
    <script>...</script>
</body>
```

- 寫成外部檔案(xx.js)

```
<head>
    <script src="xx.js"></script>
</head>
-----------------
<head>
    <script>…</script>
</head>
```

Syntax Basic

撰寫任何程式的原則：

- 易於維護
- 具可攜性(portable)
- 最好程式碼本身就能代表其意義
- 並適時加入註解(comment)，增加程式的可讀性
  - 單行註解　　// ...
  - 多行註解　　/* ...
    ...
    */
- 常見錯誤：
  大小寫字母、拼錯字、漏寫 " " 或 ' '

1. 變數

- 意義：
  - 代表記憶體空間，可存放任何數值或物件
  - 其所代表的值可任意改變
- 命名原則：
  - 由字母大小寫、數字、**$**及底線組成
  - 數字不能在開頭
  - 大小寫視為不同(case-sensitive)
  - 不能與**關鍵字(以及未來保留字)**相同(下一頁)
- 宣告時，可使用關鍵字 var,let,const
- 敘述(statement)結束，請以『；』結尾

關鍵字或保留字：

break	delete	function	return	typeof
case	do	if	switch	var
catch	else	in	this	void
continue	finally	instanceof	throw	while
default	for	new	try	with

未來保留字：

abstract	double	implements	private	throws
boolean	enum	import	protected	transient
byte	export	int	public	volatile
char	extends	interface	short	
class	final	long	static	
const	float	native	super	
debugger	goto	package	synchronized	

2.資料型態

- Javascript 有幾個重要的資料型態：
  - 數字 Number
  - 陣列 Array
  - 字串 String
  - 布林 Boolean
  - 日期時間 Date
  - 數學 Math
  - 正規表示式 RegExp
  - 物件 Object

- 也是資料型態：
  - null(空值)
  - undefined(未定義)

    宣告一個變數，沒立刻給值

    使用一個物件，但未聲明屬性

    定義了函數的參數，但並未傳值給該函數

◦宣告變數時，無須宣告資料型態，JS會在執行時自動轉換。

◦與變數有關的運算子：typeof

typeof *物件*

//顯示此*物件*是何種資料型態

//若 typeof *某函數*

//會顯示*function*，函數實際上就是物件

//若結果為null，常常是表示物件不存在，所以null會被當成物件

◦資料型態轉換

parser會視程式的前後文決定資料型態的轉換。

或使用強制轉型的語法：

```
Number(object)
parseInt(object)
object.toString()
```

- 變數的關鍵字 var, let, const
  - 宣告變數時，不一定要加上這些關鍵字。
  - 宣告變數時，盡可能立刻給初值，否則是undefined
  - 每個變數都有其scope和 life cycle
  - 其scope可大概分成區域和全域
  - 較好的程式，盡可能使用區域變數

- 定義區域變數：
  - 在函數中使用變數關鍵字
  - 定義函數時的參數

- 定義全域變數：
  - 在函數中**不要**使用變數關鍵字
  - 寫在函數之外

舉例來說：

◦ meg在此為區域變數

```
function doFirst(){
   let meg = 'error';
   alert(meg);
}
```

◦ meg在此為全域變數

```
let meg = 'error';
function doFirst(){
   alert(meg);
}
```

◦ meg在此為全域變數

```
function doFirst(){
   meg = 'error';
   alert(meg);
}
```

◦ meg在此為區域變數

```
doFirst('error');
function doFirst(meg){
   alert(meg);
}
```

3. 輸出 / 輸入

- 輸出到畫面(少用)

```
document.write('Hello World!');
```

- 跳出視窗
  - alert 視窗
  - confirm 視窗
  - prompt 視窗

- Javascript console

```
console.log('Hello World!');
```

4. 運算子、運算式與敘述

·JS的運算子

運算子優先順序	描述	結合性
()、[]		從左到右
++、--、+(正) 、-(負) 、!	一元運算子	從右到左
*、/、%	算術運算子	從左到右
+(加)、-(減)	[同上]	從左到右
>、>=、<、<=	關係運算子	從左到右
==、!=、===、!==	[同上]	從左到右
&&	邏輯運算子	從左到右
||	[同上]	從左到右
?:	條件運算子	從左到右
=、+=、-=、*=、/=、%=、...	指定運算子	從右到左

·字串運算子：字串可用"+"將字串串接

5. 迴圈敘述

- 結構化的程式設計有三種結構：
  - 循序式：即逐行執行
  - 迴圈式：反覆執行到不想執行為止
  - 選擇式：多重選項，擇一而行
- 常見的迴圈形式：

迴圈	執行次數	使用時機
while(條件判斷){敘述;}	0~N	不固定的執行次數與狀態
for(初值;條件判斷;計次){敘述;} for(...in 陣列或物件){敘述;}	0~N	固定的執行狀態
do{敘述;} while(條件判斷);	1~N	執行起碼一次

- 迴圈的好幫手break和continue
  - break;敘述除了在迴圈有作用以外，還可使用在switch
  - continue;敘述只在迴圈有作用

6. 選擇性敘述

選擇性敘述	說明	執行
if(條件){...}	·執行0或1次	符合條件就執行

if(條件){
(1)
}else{
(2)
}
	·條件大多是在一個數值範圍內；且可以放浮點數的比較	符合條件就執行(1)，
		不符合就執行(2)

switch(比對條件)
{
case XX:
...
break;
case OO:
...
break;
...
default;
} //end of switch
	·比對條件控制在整數形態或字元形態
	
	·不適合使用條件在一個範圍內，例如1~1000之間
		多重選擇

7．方法(函數)

內建方法(函數)--直接使用即可，不需要宣告

內建函數	說明
parseInt(value,base)	傳回 string 轉成 10 進制後的整數； base 不一定要寫，default 是10 進制。
parseFloat(value)	傳回 string 轉成浮點數後的數目
isNaN(testValue)	檢查 testValue 是否為非數字型態 (is Not a Number)
escape(string)	傳回string 的 16 進制編碼得出來的字串
unescape(string)	傳回以 16 進制編碼字串的原本字串
eval(string)	把 string 當作 JavaScript 的敘述來執行
Number(object)	將物件轉換成數值，遇到不合法傳回NaN

◦自訂函數--

◦宣告　◦定義　◦呼叫

◦JavaScript不用明顯的宣告，直接會寫『定義』，實際上也有『宣告』的意味

·定義

```
function functionName() {
        // 敘述
}
```

·呼叫

```
functionName();
```

有傳回值的函數：

·定義

```
function functionName() {
        // 敘述
  return value;
}
```

·呼叫

```
variable = functionName();
```

帶有參數的函數：

·定義

```
function functionName(arg1,arg2,...) {
        // 敘述
}
```

·呼叫

```
functionName(value1,value2,...);
```

Object-Oriented

1.物件的屬性與方法

- 所有能描述的東西都是由**物件**組成
  - 大的物件裡面可以包含小物件
  - 許多的小物件可以組成大物件

- 每一個物件都有其**特性**與其**使用方法**
  - 物件的特性即為**屬性(attributes)** -- 名詞 & 形容詞
  - 物件的使用方法即為**方法(methods)** -- 動詞

- 物件有層級

- **事件(events)**的發生可以啟動物件的方法或屬性

2.事件(events)處理--1/4

Javascript 是 event-driven language

--事件--

- 系統會將每個發生的動作轉成事件，然後送到程式處理

- 我們就根據不同的事件來設計不同的工作，
  負責處理事件的就叫做事件處理程序(Event Handler)

- JavaScript 的事件通常與物件有關，所以不同的物件就支援不同的事件處理

2.事件(events)聆聽功能--2/4

(1).建立事件聆聽功能

包括一個**物件**、一個**事件**、一個**處理函數**

1.嵌入式事件處理(.html)

```
<body onload="doFirst()">
```

2.過去的事件處理方法(.js)

```
window.onload = doFirst;
```

或 `window.onload = function(){...}`

3.`window.addEventListener('load',doFirst);`

2.事件(events)分類--3/4

(2).事件分類

- 輸入裝置
- 鍵盤
- Browser
- Form

2.事件(events)分類--4/4

(3).事件物件的屬性和方法

屬性：

方法：

```
preventDefault()
stopPropagation()
```

(4).引用事件物件

3. 視窗(window)物件 -- 屬性

屬性	
navigator	
location	
history	
document	
[event]	物件的[屬性和事件]本身也可以是物件

3.視窗(window)物件 -- 方法

方法		
alert()	警告訊息	
confirm()	確認訊息	
prompt()	提示訊息	
setTimeoout()	設定計時器	
clearTimeout()	取消計時器	
setInterval()	設定計時器	
clearInterval()	取消計時器	

視窗物件--window.navigator

- 提供瀏覽器的訊息

屬性	Result
appCodeName	Mozilla
appName	Netscape
appVersion	5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.111 Safari/537.36
cookieEnabled	true
language	zh-tw
onLine	true
userAgent	Win32
geolocation	
方法	
javaEnabled()	判斷是否支援JAVA

視窗物件--window.history

- 儲存該網頁曾到訪過的URL資料

屬性		
length	到訪過URL的個數	history.length
方法		
go()	載入指定的URL	history.go(number|URL)
back()	載入上一個URL	history.back()
forward()	載入下一個URL	history.forward()
pushState()		
replaceState()		

視窗物件--window.location

◦提供了很多讀取所在網址的相關資訊

屬性		
hash	取得anchor	
host		
hostname		
href		
origin		
pathname		
port		
protocol	通訊協定，包括冒號	
search	頁面跳轉	
方法		
assign()		
reload()		
replace()		載入新的 URL

視窗物件--window.document--1/3

屬性	
head	
scripts	
title	
body	
cookie	
forms	
images	
links	
files	

視窗物件--window.document--2/3

方法(JS與HTML的elements產生關聯)	
getElementById()	querySelector()
getElementsByName()	querySelectorAll()
getElementsByTagName()	
getElementsByClassName()	
修改HTML的elements的屬性與方法	
element.**innerHTML** | element.**innerText**	
element.setAttribute(attr,value) 傳回值(屬性值) = element.getAttribute(attr)	
element.style.property =	

視窗物件--window.document--3/3

方法 (新增或刪除HTML的elements)	
createElement()	
removeChild()	
appendChild()	
replaceChild()	
增加事件處理	
document.getElementById(*id*).onclick=function(){}	
document.getElementById(*id*).addEventListener(事件,處理函數)	

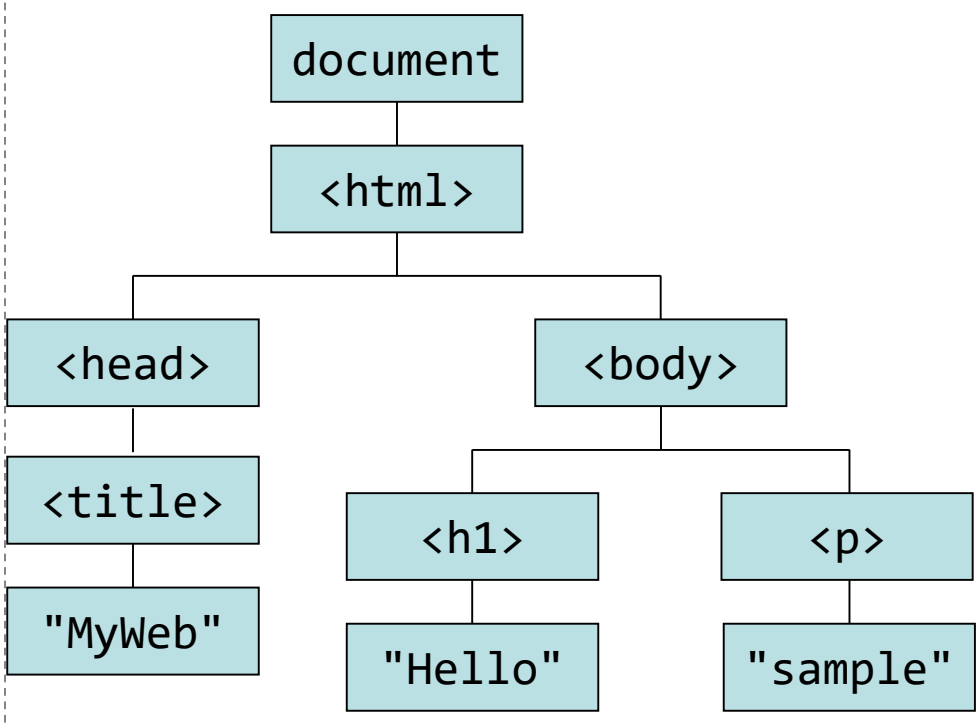
文件物件模型(DOM)--1/4

- 何為DOM?
  - DOM == Document Object Model
  - 此為W3C制定的標準
  - 基本上，DOM屬於JavaScript的介面標準，各家瀏覽器依此標準建置實際的物件，使JavaScript可以直接使用

- 基本概念
  - 節點node：文件中任何一樣東西都是一個節點
  - document node：整篇文件
  - 層次：節點彼此之間存在著層次的關係

```
<p>sample</p>
<p>標籤為一個元素節點(element node)，
並且包含文字節點(text node)sample
<p>節點則為sample節點的parent node
```

文件物件模型(DOM)--2/4



說明：

- 左側為HTML文件的部分
- 左邊10個框框都是節點
- 三個文字節點(text node)，都沒有後續節點稱為leaf
- 六個標籤皆為element node
- document為此family tree的根節點(root)
- <head>和<body>有同個父節點(parent node)，所以他們彼此之間為兄弟節點(sibling node)

文件物件模型(DOM)--3/4

- **DOM的屬性**

描述節點本身的訊息	
nodeName	- element node就是標籤名稱，一定要大寫， nodeName=='BODY' - 屬性節點的nodeName就是屬性名稱
nodeValue	
nodeType	-代表節點類型，用數字來表示 -element(1)、attribute(2)、text(3)、comment(8)、document(9)
描述彼此之間的關係	
parentNode | childNodes	
firstChild | lastChild	
previousSibling | nextSibling	
描述文字節點	
innerHTML | innerText | textContent	

文件物件模型(DOM)--4/4

- **DOM的方法**

新增	
createElement(‘標籤’)	
appendChild()	
insertBefore()	
replaceChild()	
cloneNode()	
刪除	
removeChild()	
判斷	
hasChildNodes()	判斷是否有子節點
hasAttributes()	判斷是否有該屬性

document.cookie--1/3

- 目前常用的是 web storage(HTML5)

- HTTP是一種無狀態的協定，所以web server不會記住任何東西
  假設是FB的server，有些資訊應該記住，例如會員是否已登入? 是否投過票? 是否玩過Bubble Witch Saga?就需要引入cookie
- cookie是儲存在user端的文字檔，它會與server端的目錄相關聯，當user發出request的時候，cookie會隨著HTTP的request傳送出去，而server端會決定如何處理cookie傳來的資訊
- 何處使用 Cookie：
  - 月曆記事簿
  - 登入，sticky form
  - 個人化網站

document.cookie--2/3

- 設定**cookie**

```
document.cookie ='cookie=yes; expires=Tue, 16 Oct 2012 08:30:20 UTC; path=/; domain=iii.org’;
document.cookie ='testcookie=maybe; expires=Mon, 22 Oct 2012 16:30:20 UTC; path=/; domain=iii.org';
```

參數	說明
name=value	name 是 Cookie 的名稱，value 是該 Cookie 的值， 這個字串會以 16 進制編碼。 必需要有的參數。
expires=date	指定Cookie 的有效日期，當過了有效日期後，此Cookie 就不會再儲存在瀏覽器；date是GMT的格式。如果未指定這個參數，此Cookie的有效日期就是使用者退出瀏覽器時
path=path	指定可以存取該 Cookie 的路徑。如果不指定這個參數，path就為設定該 Cookie 的網頁所在的路徑
domain=domain	指定可以存取該 Cookie 的網域。 如果不指定這個參數，domain 就指定為設定該 Cookie 的網頁所在的網域
secure	指定 Cookie 只可以傳送給 HTTPS 伺服器

document.cookie--3/3

- **讀取cookie**

  讀取先前設定的cookie，可用for迴圈將前述的字串，用『;』來當成分隔的符號，可將cookie的4或5個參數一一讀取出來

- **刪除cookie**

  新增一個有效期間為過去時間的同名cookie，用來刪除cookie

- 可用屬性navigator.cookieEnabled來看看瀏覽器是否支援cookie

```
if(navigator.cookieEnabled){
    //瀏覽器有支援，可以設定或讀取cookie了
}
```

4. 內建物件

- JavaScript 的內建物件(資料型態|類別)：
  1) Boolean
  2) Number
  3) String
  4) Array
  5) Math
  6) Date
  7) RegExp
  8) Object
  9) …

內建物件--Boolean

方法	
toString()	以字串表示布林值
valueOf()	取得物件原來的布林值

	var bool = true;
	var bool = new Boolean();
	var bool = new Boolean(true);
	var bool = new Boolean(5 > 3);

內建物件--Number

方法	
toExponential(x)	
toFixed(x)	
toPrecision(x)	
toString()	以字串表示布林值
valueOf()	取得物件原來的布林值
屬性	
MAX_VALUE	
MIN_VALUE	
POSITIVE_INFINITY	
NEGATIVE_INFINITY	
NaN	

內建物件--String

方法	
charAt()	
indexOf() | lastIndexOf()	
slice(index x, index y)	
split()	
substr(indax x, int count)	
toLowerCase()| toUpperCase()	
concat()	
replace()	
toString()	
trim()	
屬性	
length	

內建物件--Array

方法	
indexOf() | lastIndexOf()	
pop() | push(item)	
shift() |unshift(item)	
splice()	
concat()	
slice()	
concat()	
join()	
reverse()	
sort()	
屬性	
length	

內建物件--Math

- 目的：提供複雜或特殊的數學運算
- **不可以使用 new 來給初值**

- 屬性(所有數學屬性皆不可修改)
  - Math.PI
  - Math.E
  - Math.LN
  - Math.LN1
  - Math.LOG2E
  - Math.LOG10E
  - Math.SQRT1_2
  - Math.SQRT2

·方法
- Math.abs(x)
- Math.acos(x)
- Math.asin(x)
- ...
- Math.ceil(x)
- Math.cos(x)
- Math.exp(x)
- Math.floor(x)
- Math.log(x)
- Math.max(...)
- Math.min(...)
- Math.pow(x,y)
- Math.random()
- Math.round(x)
- Math.sqrt(x)

內建物件--Date--1/3

方法	說明
Wed Feb 24 2022 11:49:18 GMT+0800 (台北標準時間)	
getDate() | getUTCDate()	24
getDay() | getUTCDay()	3 (一星期的第幾天，周日為0)
getFullYear()	2022
getHours()	11 (0~23)
getMilliseconds()	
getMinutes()	49
getMonth()	1 (0代表1月)
getSeconds()	18
getTime()	從UNIX時間戳記至目前的總毫秒數

內建物件--Date--2/3

方法	
Wed Feb 24 2022 11:54:59 GMT+0800 (台北標準時間)	
toDateString()	Wed Feb 24 2022
toISOString()	2022-2-24T03:54:59.889Z
toJSON()	2022-2-24T03:54:59.889Z
toLocaleDateString()	2022/2/24
toLocaleString()	2022/2/24 上午11:54:59
toLocaleTimeString()	上午11:54:59
toString() | toUTCString	Wed Feb 24 2022 11:54:59 GMT+0800 (台北標準時間)
toTimeString()	11:54:59 GMT+0800 (台北標準時間)

內建物件--Date--3/3

更改時間的方法	
setDate()	
setFullYear()	
setHours()	
setMilliseconds()	
setMinutes()	
setMonth()	0代表1月
setSeconds()	

內建物件--RegExp

方法	
compile()	
exec()	
test()	
屬性	
global	
ignoreCase	
lastIndex	
multiline	
source	

5.自訂物件

- 目的：產生與定義自訂物件
- 建立物件：

```
let obj = new Object();
let obj= {};

let obj = new Object(name:'Anan',age:10);
let obj = {name:'Anan',age:10};
```

- [ex]

```
let obj = {
     name:'Anan',
     age:10,
     favoriteColors:{'blue', 'green', 'gray'},
     gender:'M',
     student:true,
     today:new Date(),
};
```

簡介ES6

1.var | let | const

ES5 => var(variable) 宣告的變數可以改變
ES6 => let　　宣告的變數可以改變
　　const　　宣告的變數不可以改變(不能被reassigned)

var 和 let 的差別在 #作用域(scope)

```
for(let j = 0; j < 5; j++){
   console.log(j);
}
console.log(j);
```

#hosting (提升)
JS可以補宣告

2.object shorthand 物件縮寫

(1).屬性縮寫

(2).函數縮寫

3.destructuring assignment 解構賦值.

目的:便利的取出元素。

(1).陣列解構

(2).物件解構

3.destructuring assignment 解構賦值..

(3).方法(函數)參數的解構

4.string template 字串模板.

(1).字串串接

(2).插入表達式

4.string template 字串模板..

(3).多行字串

5.arrow function 箭頭函數.

(1).宣告方式

(2).語法簡短

5.arrow function 箭頭函數..

(3).自動綁定

(4).this

jQuery

write less, do more

簡介jQuery

JavaScript的問題

- 過去，不同的瀏覽器會用不同的方式處理JS
  因此我們常常使用Firefox來撰寫JS，
  而用IE來測試程式。
- 於是將JS程式碼寫好，包裝成函數，讓使用者使用，如此可以解決瀏覽器相容的問題，並簡化了撰寫JS的工作。
- JS的函式庫：
  http://w3techs.com/technologies/overview/javascript_library/all
  https://en.wikipedia.org/wiki/List_of_JavaScript_libraries

jQuery

- 由John Resig在2006年釋出第一版
- 獲得各大公司的支援
- 輕量的Javascript Library
- 簡化了Javascript很多的例行程式
- 支援HTML DOM、CSS、Ajax …
- 更多的特效與動畫
- 方便撰寫Plugin來增加功能
- 總括來說，jQuery的優勢有：
  跨瀏覽器、相容與擴充性高、能應用web端的知識

Javascript的例行工作

- 與網頁元素產生關聯

  ```
  document.getElementById('某id');
  ```

- 動態增加網頁內容

  ```
  document.createElement('標籤');
  ```

- 修改網頁元素的屬性

  ```
  選取物件.屬性 = '更新的屬性值';
  ```

- 選取表單欄位的值

  ```
  let 變數 = 選取表單的物件.value;
  ```

- 針對事件的發生，設定處理函數

  ```
  選取物件.addEventListener('事件',函數);
  ```

Javascript與jQuery

[Javascript]

```
var pTag=document.getElementsByTagName('p');
for(var i=0; i<pTag.length; i++){
        pTag[i].style.color = 'red';
}
```

[jQuery]

```
$('p').css('color', 'red');
```

jQuery的下載與安裝

jQuery的library

- https://jquery.com/
- download jQuery(目前)--v3.6.0
  之前的版本為(v1.12.4 or v2.2.4)
  -提供兩種檔案
  - 正常版(uncompressed)：readable
    用來開發和除錯用
  - 壓縮版(compressed|minified)：可讀性低
    上線時可用，因為檔案小

CDN Hosted jQuery

CDN(Content Distribution/Delivery Network)

內容傳遞網路

https://cdnjs.com/

jQuery準備事項

- 所有的效果，理當要等所有HTML程式碼下載完之後才可以進行操作。
- 寫法：

```
jQuery(document).ready(function(){
      ……
});
-------------------------
$(document).ready(function(){
      ……
});
-------------------------
$(function(){
      ……
});
```

jQuery的語法

$(選取內容).處理方式( );

DOM物件與jQuery物件

- **DOM 物件：**

  `getElementById()`

  `getElementsByTag()`

  `getElementsByTagName()`

  `getElementsByClassName()`

- **jQuery 物件：**

  用jQuery框架將DOM 物件包裝起來產生一個新物件

jQuery的良好機制

- 自動化迴圈

```
$('p').css('color','red');
```

- **chaining function**

```
$('p').text('Hello').css().animate();
```

jQuery的選取內容

1 基本選擇器(Basic Selectors)

2 階層選擇器(Hierarchy Selectors)

3 屬性選擇器(Attribute Selectors)

4 基本篩選器(Basic Filters)

5 內容篩選器(Content Filters)

6 可視篩選器(Visibility Filters)

7 子元素篩選器(Child Filters)

8 表單相關篩選器(Form Filters)

9 其他篩選器(other)

10 關於this

$(選取內容[1]).處理方式( );

Basic Selectors	
*	$('*')
tagName	$('p')
.className	$('.className')
#idName	$('#idName')
selector,selector,selector…	

$(選取內容[2]).處理方式( );

Hierarchy Selectors	
[子孫選擇器] ancestor descendant 元素 子孫元素	
[子元素選擇器] parent > child 父元素 > 子元素	
[相鄰元素選擇器] prev + next 元素 + 相鄰元素	
[兄弟元素選擇器] prev ~ siblings 元素 ~ 兄弟元素	

$(選取內容[3]).處理方式( );

Attribute Selectors	
selector[name]	
selector[name = "value"]	
selector[name *= "value"]	
selector[name ^= "value"]	
selector[name $= "value"]	

$(選取內容[4]).處理方式( );

Basic Filters	
selector:**first**	
selector:**last**	
selector:**odd**	
selector:**even**	
selector:**eq**(數值)	
selector:**gt**(數值)	
selector:**lt**(數值)	

$(選取內容[5]).處理方式( );

Content Filters	
selector:**empty**	
selector1:**has**(selector2) selector1:**not**(selector2)	
selector:**contains**(string)	
selector:**parent**	

$(選取內容[6]).處理方式( );

Visibility Filters	
selector:**hidden**	
selector:**visible**	

$(選取內容[7]).處理方式( );

filters-child	
selector:**first-child**	
selector:**last-child**	
selector:**only-child**	
selector:**nth-child()**	
selector:**only-of-type**	
selector:**first-of-type**	
selector:**last-of-type**	
selector:**nth-of-type()**	
selector:**nth-last-of-type()**	

$(選取內容[8]).處理方式( );

filters-forms	
:text | input[type="text"]	
:password | input[type="password"]	
:radio | input[type="radio"]	
:checkbox | input[type="checkbox"]	
:reset | input[type="reset"]	
:submit | input[type="submit"]	
:button | input[type="button"]	
:file| input[type="file"]	
:image| input[type="image"]	
selector:**checked** | selector:**selected**	
selector:**disabled** | selector:**enabled**	
selector:**focus**	

$(選取內容[9]).處理方式( );

filters-other	
selector:**header**	
selector:**lang**(語言-國碼)	
selector:**animated**	
selector:**root**	
selector:**target**	

$(選取內容[10]).處理方式( );

this	
$(this)	

jQuery的處理方式

1 事件處理
2 其他的事件處理
3 ready事件處理
4 HTML屬性的處理
5 CSS屬性的處理
6 網頁內容的處理
7 DOM內容的處理
8 特效(Effects)處理
9 jQuery對元素的處理
10 公用函數與獨立資料的處理方法
11 Ajax的處理方法

$('選取內容').處理方式[1]( );

$('選取內容').事件(處理函數);

===> $('選取內容').事件(function(){});

http://api.jquery.com/category/events/

Events - 輸入裝置	
click	
mousedown	
mouseup	
dblclick	
mousemove	
mouseover | mouseenter	
mouseout | mouseleave	
hover	

Events - 鍵盤	
keypress	
keydown	
keyup	
Events - 瀏覽器	
$(window).resize()	
$(window).scroll()	
Events - 表單	
submit	
reset	
select	
focus	
blur	
change	

$('選取內容').處理方式2&3( );

其他事件處理	
bind('事件名稱', 處理程序)	新增事件
unbind('事件名稱')	刪除事件
$(selector).on('事件名稱',處理程序); $(document).on('事件'[,'selector',data],function(){});	對未定義元素設定事件
off('事件名稱')	將on()設定的移除
trigger('事件名稱')	
one('事件名稱',處理內容)	
ready 事件	
$(document).ready(處理內容);	
$(function(){處理內容});	

$('選取內容').處理方式4( );

處理HTML的屬性	
attr('屬性名稱')	傳回(查看)屬性值
attr('屬性名稱','屬性值')	設定屬性值
removeAttr('屬性名稱')	
addClass('class名稱')	
removeClass('class名稱')	
toggleClass('class名稱')	
hasClass('class名稱')	
處理Javascript的屬性	
prop('屬性名稱')	
removeProp('屬性名稱')	

$('選取內容').處理方式[5]( );

處理CSS的屬性	
css('css屬性名稱')	傳回(查看)屬性值
css('css屬性名稱', '屬性值')	設定屬性值
width()	
height()	
offset()	
position()	

$('選取內容').處理方式[6]( );

網頁內容	
html('HTML字串')	
text()	傳回(查看)文字節點的內容
text('字串')	設定文字節點的內容

$('選取內容').處理方式[7-1]( );

HTML DOM的操作	
append(HTML字串)	
prepend(HTML字串)	
after(HTML字串)	
before(HTML字串)	
remove()	
detach()	
replaceWith()	

$('選取內容').處理方式[7-2]( );

尋找DOM	
find() | filter()	
parent()	
children()	
first()	
last()	
prev()	
next()	
eq()	
slice()	
$(selector1).not(selector2); $(selector1).filter(selector2);	

$(選取內容).處理方式[8-1]( );

Effects - basic	
hide()	$(selector).hide(speed,callback);
show()	$(selector).show(speed,callback);
stop()	
Effects - Fading	
fadeIn(speed,callback)	
fadeOut(speed,callback)	
fadeTo(speed,opacity,callback)	

$(選取內容).處理方式[8-2]( );

Effects - Sliding	
slideDown()	$(selector).slideDown([speed,easing,callback])
slideUp()	$(selector).slideUp([speed,easing,callback])
slideToggle()	$(selector).slideToggle([speed,easing,callback])

Effects - animate
animate({properties}[, duration, easing, complete])
stop(true | false) true:可中斷執行中的動畫 false:不可中斷

$(選取內容).處理方式9( );

jQuery對元素的處理	
$('selector')	
$('HTML字串')	將HTML字串轉為jQuery字串
$('selector').each(function(){});	
$('selector').index(元素);	

$(選取內容).處理方式[10]( );

公用函數的處理方法	
$.trim('字串')	
$.isEmptyObject(物件)	
$.isFunction(物件)	
$.contains(元素1,元素2)	
$.inArray(物件,物件陣列)	
$.each(陣列,callback)	
獨立資料的處理方法	
$().data('屬性')	
$().data('屬性', '值')	
$().removeData('屬性')	

$(選取內容).處理方式[11]( );

Ajax方法
$.ajax(選項); e.g. jQuery.ajax(url[,settings]);
$.get(); e.g. jQuery.get([settings]);
$.post() e.g. jQuery.post(url[,data][,success][,dataType]);
$.getJSON() e.g. jQuery.getJSON(url[,data][,success]);
$.getScript() e.g. jQuery.getScript(url[,success]);
$(selector).**load**(同一個domain內的URL);

js事件物件(event object)

Attributes	
event.pageX	
event.pageY	
event.type	
event.target	
event.currentTarget	
event.relatedTarget	
event.which	
event.data	
Methods	
event.preventDefault()	取消預設行為
event.stopPropagation()	中斷事件的傳遞