宣告 / Declare

var	常數, 變數, 函
let	變數, 函式
const	常數, 函式

函式 / Function (copy)

函式宣告數(Function Declaration)	function A(參數) { 定義 }	不管定義寫在哪，執行時都會被移到最前面，因此到處都可以呼叫，不用管寫程式時的先後順序。
具名函式表達式(Function Expression)	let a = function A(參數) { 定義 };	須在前面定義，後面才能呼叫。函式名稱只有在此函式的定義內有效，很沒用。
匿名函式表達式(Function Expression)	let a = function(參數) { 定義 };	須在前面定義，後面才能呼叫。
立即函式(Immediately-Invoked Function Expression/IIFE)	(function(參數){ 定義 })(參數);	會立刻執行，適合將 Init 的東西寫在裡面，再放在適當位置。
箭頭函式(Arrow Function)	let a = (參數) => {定義};	函式裡的this，是指向定義時的對象，而不是運行時的對象。也可以說箭頭函式內部，並不會配置一個自己的this。

函式參數 / Function Parameter

少傳的參數	自動填入 undefined
多傳的參數	自動忽略
參數預設值(Default parameter)	function A(name = 'Mark') {}
其餘參數(Rest Parameter)	function A(...args) {}

函式可以當成參數傳入

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

設定函式的 this 對象

函式.call(對象, 參數1, 參數2, ...)	適用情境：確定參數數量的時候，立刻執行。
函式.apply(對象, 參數陣列)	適用情境：不確定參數數量的時候，立刻執行。
函式. bind(對象, 參數1, 參數2, ...)	適用情境：想得到一個設定後函式
函式. bind(對象, 參數1, 參數2, ...)()	加上()可以讓回傳的參數立刻執行

[備註]
call 立刻執行
apply 立刻執行
bind 回傳一個設定後的函式

[參考資料]
【优雅代码】深入浅出 妙用Javascript中apply、call、bind
https://www.cnblogs.com/coco1s/p/4833199.html

轉型 / Type Casting

數字.toString()	數字轉字串
Sring(數字)	數字轉字串
Number(字串)	字串轉數字
parseInt(字串)	字串轉數字
parseFloat(字串)	字串轉浮點數

字串

單引號	let a = 'Hi';
雙引號	let a = "Hi";
多行字串	let a = `Hi`;
樣板字串	let b = 10; let a = `Hi ${b}`;
字串.length	取得字串長度
字串.trim()	去空白
字串.indexOf(搜尋的字串, 搜尋起始索引)	搜尋，找不到會回傳-1，找到會回傳第一個符合的位置。(第二個參數非必要)
字串.slice(起始索引, 結束索引)	子字串，索引從0開始，不包含結束索引的字元。
字串.replace(原字串, 新字串)	取代所有符合的字串
字串.split(切割詞)	分割字串
字串.toLowerCase()	轉小寫
字串.toUpperCase()	轉大寫

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

陣列 / Array	
宣告	let a = [];
遍歷	a.forEach(function() { });
陣列序列化(逗號分隔)	let c =a.toString();
合併陣列內元素	let c = a.join(合併符號)
合併多個陣列	let c = a.concat(b);
插入/刪除元素(return: 被刪除的值)	let c = a.splice(起始索引, 取代數量/可以是0, 元素/非必要);
新增元素(頭)	a.unshift(元素);
新增元素(尾)	a.push(元素);
取得第一個元素(並移除)	let c = a.shift();
取得最後一個元素(並移除)	let c = a.pop();
反轉陣列	let c = a.reverse();
切割陣列	let c = a.slice(頭, 尾);
排序	let c = a.sort();
是否包含(true/false)	let c =a.includes(值);
至少有一個元素符合(true/false)	let c =a.some(函數);
所有元素都必須符合(true/false)	let c =a.every(函數);
符合的第一個元素(return: undefied/值)	let c =a.find(函數);
符合的所有元素(return: 陣列)	let c =a.filter(函數);
加總	let c = a.reduce(初始值/加總值, 當前值);
得到處理後的新陣列	let c = a.map(函數);

陣列內容不限定單一型態

[範例] 找出偶數
let a = [0,1,2,3,5,6,7,8,9];
let c = a.filter(x => x%2 == 0);

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com