Day1

1. 未宣告 var 的變數，都會自動變成全域變數
2. const、let 範圍只在大括號裡，脫離大括號就娶不到，const定義後 就不能在修改
3. const 在 物件 陣列，可以增加東西，不能改變型別
4. let 可以改變型別
5. 變數本身不會帶有資料型別的資訊，變數沒有型別，值才有
6. 再執行時期，透過變數內容才得知變數型別
7. 可以用 \ 跳脫字元
8. NAN 跟任何數字運算都是 NAN，NAN不等於任何數字，甚至是自己
9. 可以用 isNAN(value) 來判斷是否為NAN
10. var a // undefined ，尚未給值，未定義
11. var b = null // null 明確代表此變數沒有值
12. null 、 undefined 透過Boolean() 會代表 false
13. 一個物件可以是零至多種組合，屬性是 key 與 value的組合，一個屬性的值可以是某個型別，也可以是物件，也可以是函數
14. 建立物件 var obj = new Object() 或者 var obj = { };
15. 物件取值 obj.key 或只者 obj [“{key}”]，當key 帶有空白的字串或數字，則用第二個方式
16. 屬性增加，直接obj.key = value
17. 屬性刪除， delete obj.key
18. 檢查物件屬性可以透過 key in obj 或者 obj.hasOwnProperty(‘key’) 都回傳true / false
19. 陣列是特別的物件，有順序性的集合，只能透過[ ]加上索引來存取
20. 陣列可以透過改變length改變陣列長度
21. 同一個陣列可以使用多種資料型別
22. 陣列末端新增元素，可以用 psuh() 方法，並返回新的長度
23. 陣列前段新增元素，可以用 unshift()方法，並返回新的長度
24. 刪除最後一個元素，可以用 pop()方法，並傳回被刪除的元素
25. 刪除第一個元素，可以用 shift()方法 ，並傳回被刪除的元素
26. 數字與undefined 相加時 123 + undefined // NaN
27. 數字與null 相加時 123 +null // 123 ， null會被嘗試轉成數字 也就是0
28. 數字與boolean相加時 true會轉成 1 ，false 會轉成 0
29. Number(string) 會將字串轉成數字 ， 或者 + - 在 string 前面也能轉成數字
30. 兩個 == 會做自動轉型， 三個 === 不會替數值做自動轉型
31. != 跟 !== 前面會自動轉型 後面 不會

Day 2

1. 當兩個物件比較時，如果指向同一個實體會回傳true，否者 false
2. 比較運算中，當一個值是 Boolean 時，true 會看成1，false 會看成 0
3. 陷阱 var a = b = 10，等於 var a = b，b = 10，b沒有var宣告，會是全域

變數。

1. 經過Boolean()會轉成 false 結果

* Undefined
* Null
* 0,+0 ,-0,Nan
* 空字串””或’’

1. Console.log(a || b) 對 || 運算式來說 若第一個數值為 true 則回傳第一個數值

Console.log(a && b) 對 && 運算式來說 若第一個數值為 true 則回第二個數值

在對回傳的數值 去做 boolean() 判斷

1. 所有資料可以透過 Boolean 轉成 true 或 false
2. switch(運算式，變數，值){

case : switch()內的結果

break

default

break

}

1. 在 for，while 迴圈中，break直接離開迴圈，continue則是跳過這一次繼續下一次
2. Math.ceil 無條件進位 ，Math.floor 無條件捨去 ，Math.random() 0~1之間隨機小數
3. 宣告方法

* 函式宣告 function 名稱 (參數1,參數2){ 方法內容 }
* 函式運算式 var 名稱 = function (參數) { 內容 }
* var 名稱 = (參數)=>{ 內容 }

1. 函數宣告 跟函式運算式差別在於，函式宣告可以在定義函式之前呼叫，而函式運算式則不可以，會出現錯誤。
2. test(); function test(){return 1}; 呼叫函式在定義函式之前可以
3. test(); var test = function(){return 1}; 呼叫函式在定義函式之前不可以
4. 切分變數有效範圍的最小單位式 ”function”
5. var s = 1 ， function test (){var s=100}，外面的s 與函式裡面宣告的s 是不同的變數
6. 當 function 內部沒有宣告變數時，會往外層去找， var s = 1 ， function test (){s=100} 外面的s 與函式裡面宣告的s 是同的變數
7. function 可以讀取外層宣告的變數 ，但外層拿不到裏面的變數
8. 即使寫在函式內，沒有宣告var 的變數會變成全域變數
9. 全域變數指的是全域物件的屬性，Ex: window.變數名稱
10. function test(console.log(x); var x = 1) 結果會是undefined ，因為function 裡面有宣告變數，雖然在下面，但會把變數提升到 console.log() 前面，這種就叫變數提升,let 宣告的變數不會提升
11. 使用變數前，要先宣告，是比較好的習慣
12. 基本型別，當我們在判斷這兩個變數是否相等，看的是變數裡面的值

* Var a=b=10 ，當改b時，a不會變

1. 物件型別，是傳址，

* Var a= b = {value=10} 當改b時，a會改變

1. var a={value:10}; function test (obj) { obj = { value:123 }} ; 當 a帶入 test(a); console.log(a) 還是 a={value:10} ，當obj被賦予新值時，外面的變數內容不會改變
2. var a={value:10}; function test (obj) { obj.value=123 }} ; 當 a帶入 test(a); console.log(a) 是 a={value:123}

Day3

1. BOM 是與瀏覽器 溝通的窗口，不涉及網頁內容， 核心window物件
2. DOM 是用來控制網頁的節點與內容的標準
3. 網頁被載入到瀏覽器時，瀏覽器會分析HTML檔案，由上而下依序分析，

當瀏覽器瀏覽到<head>裡的<script>內容時，會暫停分析網頁，並且立即執行<script>裡的內容

1. document 是 DOM裡的根結點，當分析HTML時都是從document開始
2. document.querySelector，document.querySelectorAll可以用CSS選擇器
3. 可以用 Node.hasChildNodes() 判斷 DOM裡 是否有子節點
4. 每個物件都有childNodes 屬性，有length屬性，可以透過[ ] 加上索引的方式來存取，但是他不是陣列，沒有陣列才有的foreach()，map() 的方法
5. Node.fristChild 可以取得 Node節點的第一個子節點，如果沒有，回傳 Null，換行，空白也算子節點，取到會是undefind
6. Node.lastChild 是 Node節點的最後一個子節點
7. Node.parentNode 取得父節點
8. Node.previousSibling，可以取得同層之間的前一個節點，如果Node 已經是第一個節點，則回傳null
9. Node.nextSibling，可以取得下一個節點，如果已經是最後一個，則回傳Null
10. Document.createElement(tagName) 可以建立一個新的元素，透過 appendChild()、insertBefore()、replaceChild() 方法將新元素加入指定的位置之後才可以看到
11. 可以對新建立的元素指定屬性 ，document.crateTextNode()建立文字節點

var newDiv = document.create(‘div’);

newDiv.id=”newDiv”;

var text = document.createTextNode(“hello world!”);

newDiv.appendChild(text)

1. 當需要大量DOM操作時，用createDocumentFragment()效能比較好
2. Node.appendChild(childNode) 加入父點節的末段
3. Node.insertBerfore(newNode,refNode) 加入到 refNode前面
4. Node.replaceChild(newChildNode,oldChildNode) 將舊元素 替換成 新元素
5. Node.removeChild(childNode)將childNode子節點移除
6. 如果要將某元素的內容全部清空，更快的方法 Node.innerHtml=’’;
7. Html 負責資料與結構
8. CSS 負責樣式與呈現
9. JavaScript 負責行為與互動
10. Node.classList.add(className)、Node.classList.remove(className) 新增或移除 class
11. 對於特殊情況要寫入CSS 樣式 可以用.

Var head = document.querySelector(‘head’);

Var link = Node.createElement(‘link’);

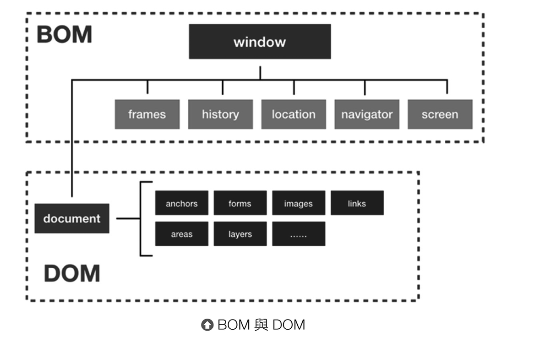
If(某些情況){

link.rel = “stylesheet”;

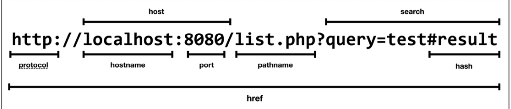
link.type = “text/css”;

link.href=”xxx.css”;

head.appendChild(link); }

1. 

透過瀏覽器的BOM window 存取 document DMO模型

1. 瀏覽器會透過一個全域物件存放所有全域變數，而全域變數都會變成全域物件的屬性。
2. 每個新的分頁，window 物件都是獨立的
3. 更換新的頁面 window.location.href =”https//www.google.com”;
4. 跳到對應的錨點 location.hash=”title”
5. 

protocol、host、hostname、port、search、hash

1. 除了 location.href 可以跳頁之外，還可以用 location.replace(href)來跳頁，差別在於 location.href會記錄在瀏覽器的歷史紀錄，可以透過上一頁會到，而location.replace(href) 不會再瀏覽器不會留下歷史紀錄，不能用上一頁返回
2. 如果要知道網頁的 href解析，可以先createElement(‘a’)出來，在透過 a.href = href，就可以 a.hostname ,a.host …等屬性可以用
3. window.history.back() 上一頁
4. window.history.forword() 下一頁

輸入法事件

1. compositionstart 事件 輸入框開始輸入法時，且正在拼字時觸發
2. compositionupdate 事件 輸入框內開啟輸入法，且正在拼字或選取字時更改了內容時觸發
3. compositionend 事件 輸入框內開啟輸入法，拼字獲選字完成，正要送出至輸入框時觸發

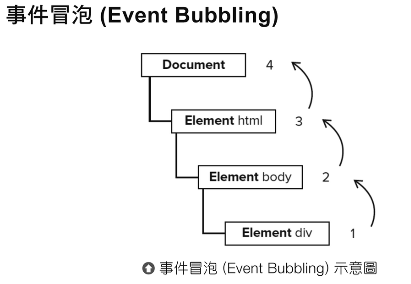
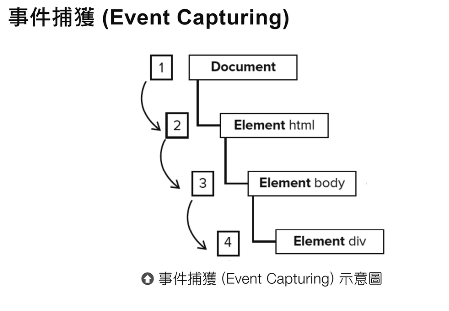
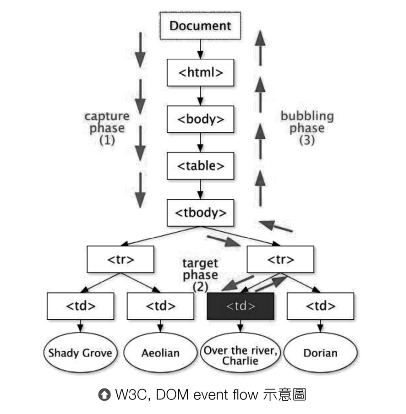
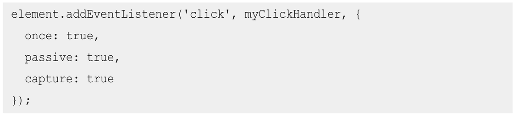
剪貼簿事件

1. cut 事件 當使用者選取某段文字，並進行剪下動作時觸發
2. copy 事件 當使用者在網頁上選取某段文字，並進行複製動作時觸發
3. paste 事件 當使用者將剪貼簿的文字貼上時觸發
4. 事件發生時，都會建立 事件物件(Event Object)，裡面包含與這個事件有關的屬性，並且以參數的行傳給我們

* event.type 事件的名稱
* event.target 觸發事件的元素
* event.bubbles 是否在冒泡階段觸發
* event.pageX/pageY 表示事件觸發時，滑鼠座標在網頁的相對位子

1. 阻擋預設行為 event.preventDefault(); ，並不會阻止事件向上傳遞(事件泡泡);
2. 阻擋冒泡事件 event.stopPropagation();， Jquery 在 event handler 最後加上 return false 來得到 preventDefault() 與 stopPropagation() 的效果，但在 JS addEventListener() 沒有作用，只有在 onclick= return false 才有作用
3. 在事件中，this 代表的是觸發事件的目標，也就是 event.currentTarget
4. 如果要在 ul 裡的 li 都綁定 click 事件 ， 如果是以 li的角度綁定事件的話，則新增新的li時可能會沒有綁定 click事件，但是如果是以ul的角度針對底下 li綁定事件，則後來新增的li 也會綁定到事件

Day 4

1. 事件冒泡指的是 從啟動事件的元素節點開始逐層往上傳遞，直到整個網頁的根節點
2. 
3. 捕獲事件則剛好相反
4. 
5. 捕獲跟冒泡都會執行，會先捕獲在冒泡
6. 
7. 先執行 capture在執行 bubbling
8. 監聽事件 addEventListener(“事件名稱”,function(){ } , true) // 第三參數 true/false 分別代表 捕獲/冒泡機制 ，不指定預設為冒泡事件
9. 另一種綁定事件，on-event ex : windw.onload= function(){ }，解除為 window.onload =””;
10. 使用 addEventListener 的好處可以對同一個元素同一個事件綁定很多處理器
11. addEventListener 要解除綁定 removeEventListener()來取消，三個參數跟 addEventListener 一樣
12. removeEventListener 與 addEventListener 的處理器必需要同一個實體，所以要先將處理器拉出來 用函示運算式來宣告
13. addEventListener ，Chorme55 之後可以傳入 option這個物件
14. 

一張含有 文字, 收據 的圖片

自動產生的描述

1. window.onload 指著是 (css js 圖片等資源)全部載入完畢才觸發，
2. img的onload 則表示圖片載入完後才觸發
3. unload、beforeunload 分別是 離開頁面 或重新整理時觸發，而beforeunload 會跳出 對話框詢問使用者是否要離開頁面
4. error 是在document或者 圖片載入錯誤時觸發，比較適合使用侵入式方式比較適合



1. fullscreenchange 當使用者為全螢幕或者還原視窗時觸發
2. scroll 當瀏覽器(window)或者指定元素的卷軸被拉動觸發
3. mousedown 滑鼠按下觸發
4. mouseup 滑鼠放開觸發
5. 滑鼠移入某元素 觸發 mouseenter/mouseover 事件
6. 滑鼠在元素內移動 觸發 mousemove 事件
7. 滑鼠離開元素 觸發 mouseleave/mouseout
8. mouseenter / mouseleave 無冒泡事件 比較晚觸發
9. mouseover / mouseout 有冒泡事件 比較早觸發
10. keydown 壓下鍵盤觸發
11. keypress 除了 Shift、Fn、CapsLock 這三種按鍵按下時會觸發，按著不放則會連續觸發
12. keyup 放開鍵盤時觸發
13. 執行順序 keydown keypress keyup
14. event.keyCode 屬性可以查詢使用者按下哪一個建



1. input 事件 當 input、textarea 當內容改變的當下觸發

一張含有 文字 的圖片

自動產生的描述

1. select 事件 在 input、textarea 元素裡 選取文字時觸發可以在 event.target裡的selectionEnd、selectionStart、value的屬性中取得

一張含有 文字 的圖片

自動產生的描述

1. change 事件 在 input、select、textarea、readio、checkbox等表單元素改變後，失去焦點觸發
2. focus 事件 表單元素被聚焦時觸發
3. blue 事件 比單元素失去焦點觸發

輸入法事件

1. 
2. 一張含有 文字 的圖片

   自動產生的描述

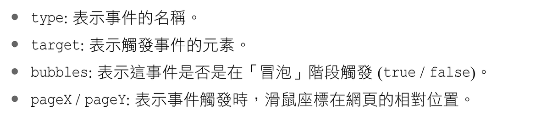
剪貼簿事件

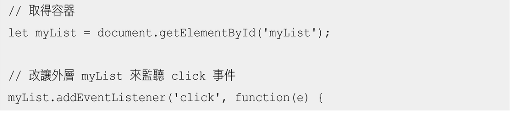
1. 一張含有 文字 的圖片

   自動產生的描述

事件 event 物件

1. 一張含有 文字 的圖片

   自動產生的描述
2. 
3. 
4. 
5. 一張含有 文字 的圖片

   自動產生的描述
6. 
7. 要對 ul 底下的li綁定 click事件時，如果以li角度去綁定事件，那新增出來的li不會綁定到 click事件，如果改成用 ul角度去綁定底下li的click事件，新增出來的li 也會被綁定
8. 
9. 一張含有 文字 的圖片

   自動產生的描述

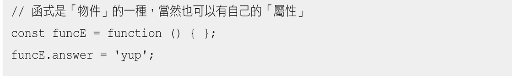
Day5

1. 可以將函示存放在 變數、陣列、物件中

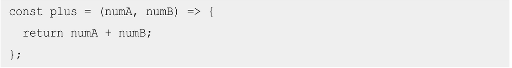
一張含有 桌 的圖片

自動產生的描述

1. 函示是物件，可以去給他屬性，可以對物件所做的事，對函數都可以做



1. 箭頭函示



1. 箭頭函示如果不需要參數，前面的小括號不可以省略
2. 而只有一個參數時，前面小誇號可以省略
3. { } 裡 如果只有一行，{ return …..} 可以省略，只留下程式碼
4. 箭頭函示沒有 arguments
5. 多傳的參數，在多數的情況下沒意義，但可以透過 arguments這個物件取得多傳的參數，他只有索引，內建length屬性，與陣列完全不同
6. 在需要執行遞迴時，可以執行 arguments.callee() 來達成
7. 再不確定參數時，可以利用 arguments物件取得所有傳遞來的參數

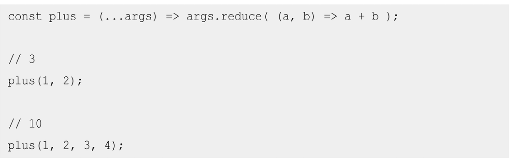
一張含有 桌 的圖片

自動產生的描述

1. 換成箭頭函數可以利用一張含有 文字 的圖片

   自動產生的描述

像上面範例，當最後一個參數以…最為開頭的命名，會被視為一個陣列，將所有參數存入陣列中，因為是陣列，可以使用arguments沒有的redcue方法來使函示變得更簡單



1. Array.reduce 作用是 第一次先傳 1、2進去，再將 結果和3再傳入函示裡

再把結果與4再傳入函示裡

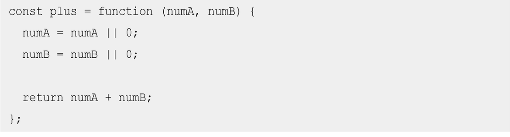
1. 當函示要傳入的參數過多的時候，可以改用物件來取代一堆函數

一張含有 桌 的圖片

自動產生的描述

好處是不用按照順序，甚至有些非必填的屬性忽略也沒問題，使著程式碼可以更容易略讀

1. 參數預設值檢查，當沒傳入參數時，函式的參數會是undefined.
2. 為了避免 undefind，可以再傳入時可以利用 ||(OR)來處理

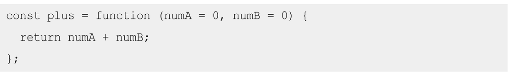


1. 會判斷成false不只 undefined 改用嚴謹一點

一張含有 桌 的圖片

自動產生的描述

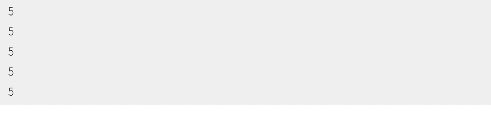
1. 或者可以替參數指定預設值



1. Callback function 指的是當滿足某個條件，函式才會被動的去執行
2. 切分變數有效範圍的最小單位式 function
3. JS 是非同步的語言

一張含有 文字 的圖片

自動產生的描述

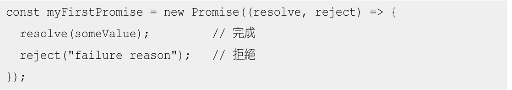


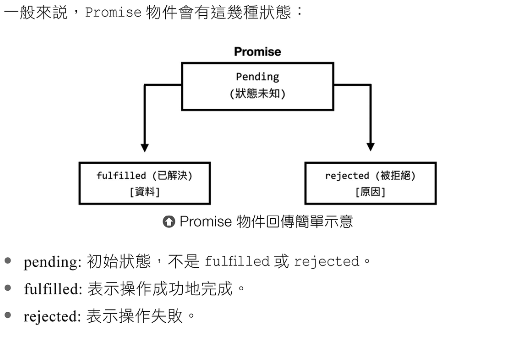
1. 立即函數 (function(傳入的參數) { } )(要傳遞的參數)
2. 解決 19的問題

一張含有 文字 的圖片

自動產生的描述

1. Promise物件，回傳的結果是 完成 或者 拒絕



1. 
2. 如果需要串聯執行多個promise功能的話，可以透過 .then()來做到

一張含有 文字 的圖片

自動產生的描述

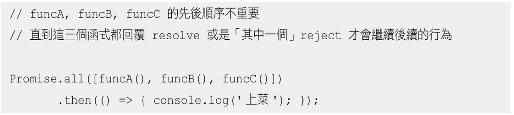
一張含有 桌 的圖片

自動產生的描述

傳遞參數要用陣列去傳 resolve([參數A，參數B]);

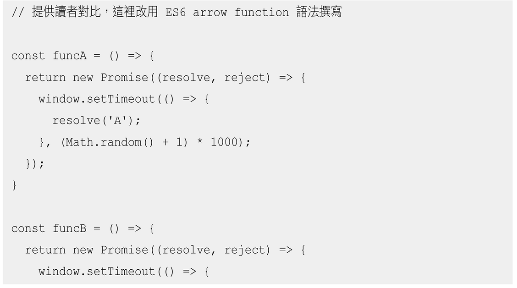
function(a) => a是參數陣列

1. 如果不在乎 funcA()、funcB()、funcC() 誰先誰後，可以用Promise.all() 與 Promise.race()
2. Promise.all() ，不管先後順序，三個都完成 或 其中一個回傳reject 才會繼續動作



要特別注意的是，當Promise.all()過程中，只要有任何一個Promise物件執行發生錯誤例外，或有 reject情況，會立刻回傳 reject狀態的 Promise物件

1. Promise.reac() 只要其中一個reslove，就會進行下一步



一張含有 文字 的圖片

自動產生的描述

1. 使用 async / await

一張含有 文字 的圖片

自動產生的描述

1. async / await 可以使用 Promise.all() 、 Promise.resc();
2. setTimeout 延遲某段時間執行一次，並回傳獨立的 timerID
3. setInterval 延遲某段時間，不斷循環，並回傳獨立的 timerID
4. 取消setTimeout 利用 clearTimeout(timerID);
5. 取消setInterval利用 clearInterval(timerID);

Day6

1. 閉包 當內部 inner 函式被回傳後，除了本身程式碼之外，已可以取得內部函式當時的環境的變數值，記住當時的環境，這就是閉包。



一張含有 桌 的圖片

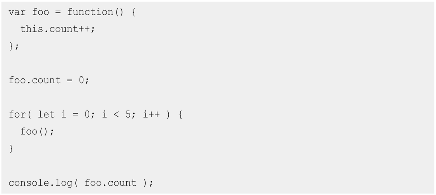
自動產生的描述

範例二

一張含有 桌 的圖片

自動產生的描述

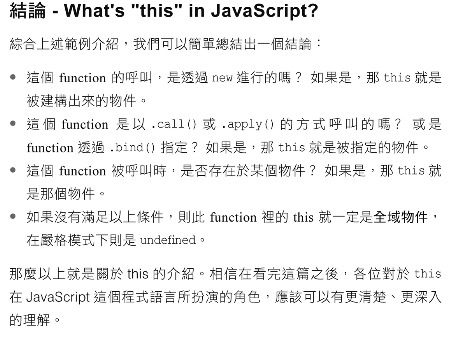
1. this 是function 執行時，自動生成的一個內部物件
2. this 代表就是呼叫 function 的物件，不等於 function
3. this 會因為執行的環境不同，與 宿主上物件的不同，而有所不同



foot 是全域變數，裡面的this 指的是window.count;

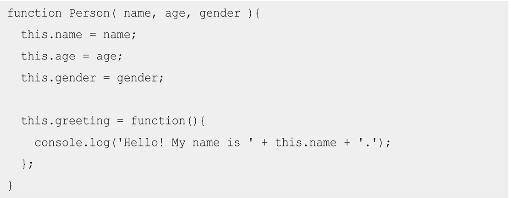
跟 foot.count 不同，所以跑迴圈時，是window.count ++

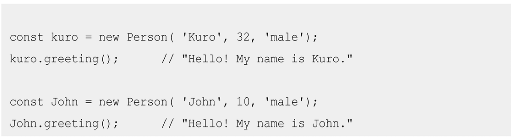
不是 foot.count++



Day7

1. 利用函式來當作建構式來建立物件





1. 透過 Object.create() 建立物件





一張含有 文字 的圖片

自動產生的描述

1. Object.create()複製出來的，去修改屬性的值，也不會影響原型物件
2. Object.create()建立出物件，就給預設值

一張含有 桌 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述



一張含有 文字 的圖片

自動產生的描述

1. 利用 Object.deFineProperty 來定義物件的屬性描述
2. 用法式 Object.deFineProperty(obj,prop,descriptor)

一張含有 文字 的圖片

自動產生的描述

Object.deFineProperty 建立 屬性為false

一張含有 文字 的圖片

自動產生的描述

物件實字建立 屬性為 true

一張含有 文字 的圖片

自動產生的描述

物件實字建立 會繼承 Object 屬性

一張含有 文字 的圖片

自動產生的描述

Object.deFineProperty 不會繼承 Object 屬性

1. 判斷閏年

一張含有 文字 的圖片

自動產生的描述

1. 物件的原型鏈與繼承，透過[原型]繼承可以讓本來沒有某個屬性的物件去存取其他物件的屬性
2. 可以透過 Object.setPrototypeO(洛克人，剪刀人) 將別的物件指定為原型

一張含有 桌 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

同一個物件無法指定兩種原型物件

一張含有 文字 的圖片

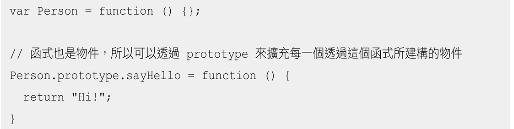
自動產生的描述

1. 原型鏈，當某個物件試著要去存取不存在的屬性時，那麼會往它的原型物件去尋找

一張含有 桌 的圖片

自動產生的描述

1. 最頂層的原型物件 Object.prototype



透過 prototype 擴充每一個透過這個函式所建構的物件

1. 只有函式才有 prototype 這個屬性
2. 要透過 new 出來的才會有 prototype的屬性
3. 當物件與他的原型有同樣的屬性或方法時，會優先存取自己的屬性或方法

沒有才會往原型去找

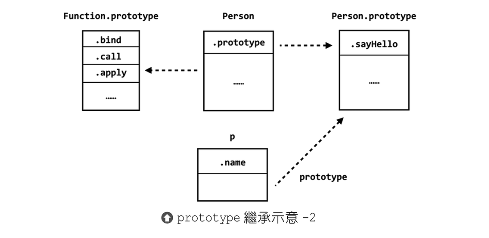
一張含有 文字, 收據 的圖片

自動產生的描述

Writable 為false 時 ，obj.name = 值時，不會被修改

一張含有 文字 的圖片

自動產生的描述



1. ES6 加入 Class

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述



一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

補充

1. 解構式

Var data = { name=”AAA” , age:12 }

以前都是 data.name 取值

現在可以改用 const {name ,age } = data;

物件key 會變成 變數

1. 如果要把data 當成另一個物件的屬性

可以直接這樣寫

const user = {

data,

address:”Taiwan”

}

就可以複製data在user物件裡

1. 箭頭函式沒有this，箭頭函數要取得觸發的元素，要透過返回的e 的target屬性來取得

( e)=>{ console.log(e.target)};

1. 用變數宣告的涵式才可以使用箭頭函式
2. 箭頭函式只有一行程式碼，可以省略 { return }
3. arr.map 會掃過陣列裡的每一個值，處理完，並回傳一個新的陣列
4. 涵式傳入的參數設定預設值，可以避免一些基本的小問題
5. 字串模板 利用 `我今年${ag參數}歲` 來組字串