

# D3.JS

## 互動式資料視覺化

**Lecturer: LinJer 林哲**

evin92@gmail.com





# 解答時間

如何使用 `Math.random()`  
產生一個介於範圍  
 $N \sim M$  之間的整數



1



2

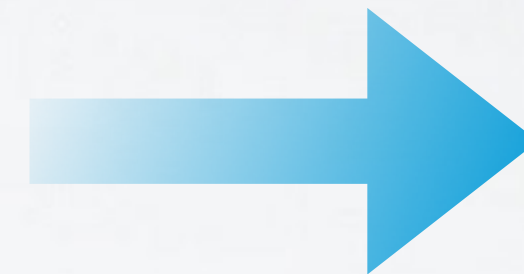
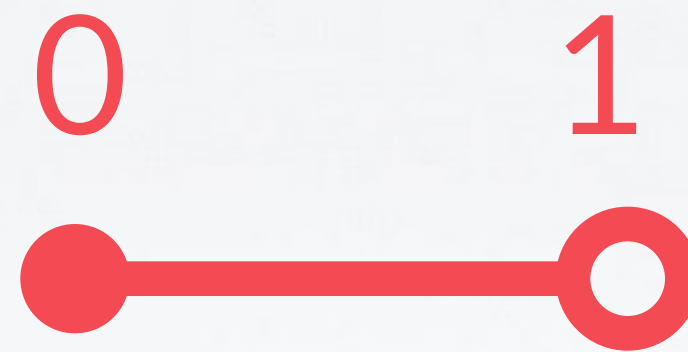




# 動腦提示

先看看如何產生1至3之間的數？

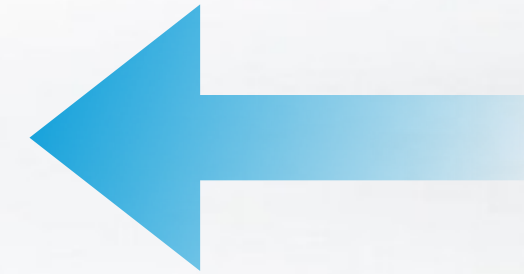
$$0 \leq \text{Math.random()} < 1$$



$$0 \leq \text{Math.random()} * 3 < 3$$

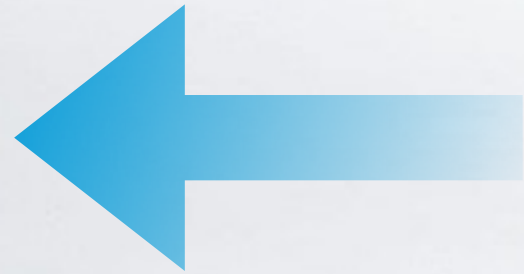
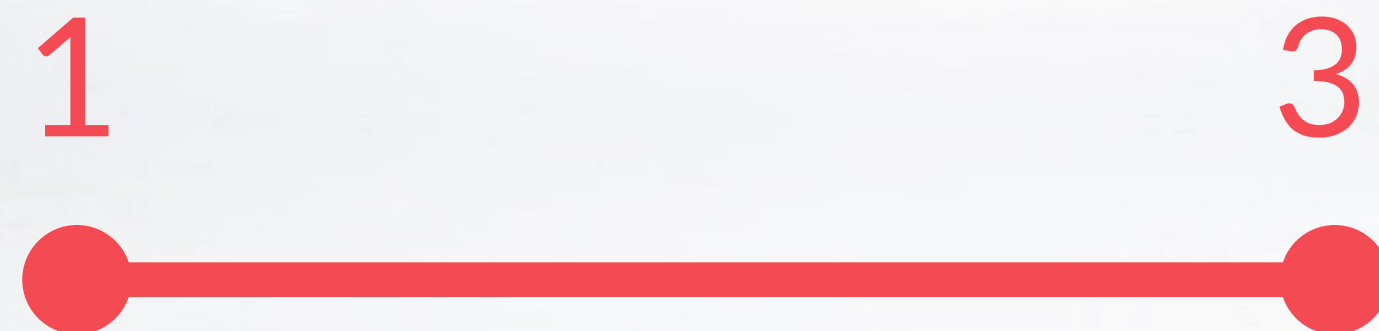


$$1 \leq \text{Math.random()} * 3 + 1 < 4$$



$$1 \leq \text{Math.floor}(\text{Math.random()} * 3 + 1) \leq 3$$

$$1 \leq \text{Math.floor}(\text{Math.random()} * 3 + 1) \leq 4$$



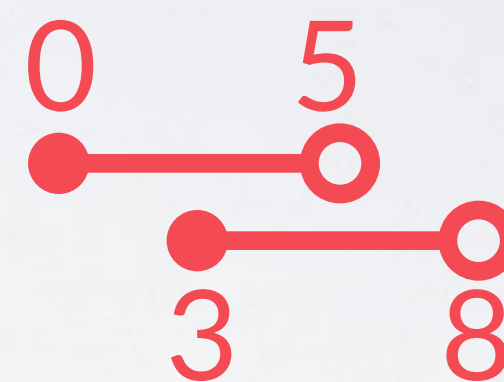
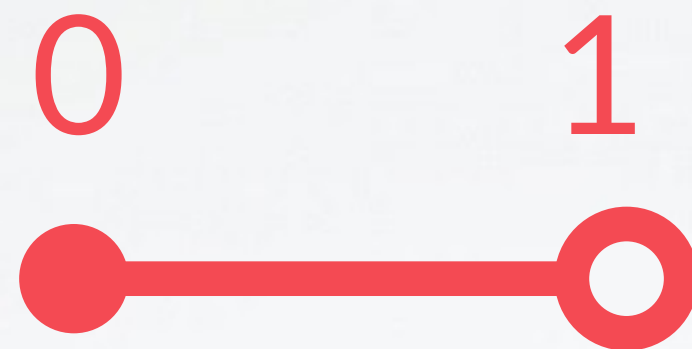
$$1 \leq \text{Math.floor}(\text{Math.random()} * M + 1) \leq M$$



# 動腦提示

如何產生3至8之間的數？

$$0 \leq \text{Math.random()} < 1$$



$$0 \leq \text{Math.random()} * 5 < 5$$

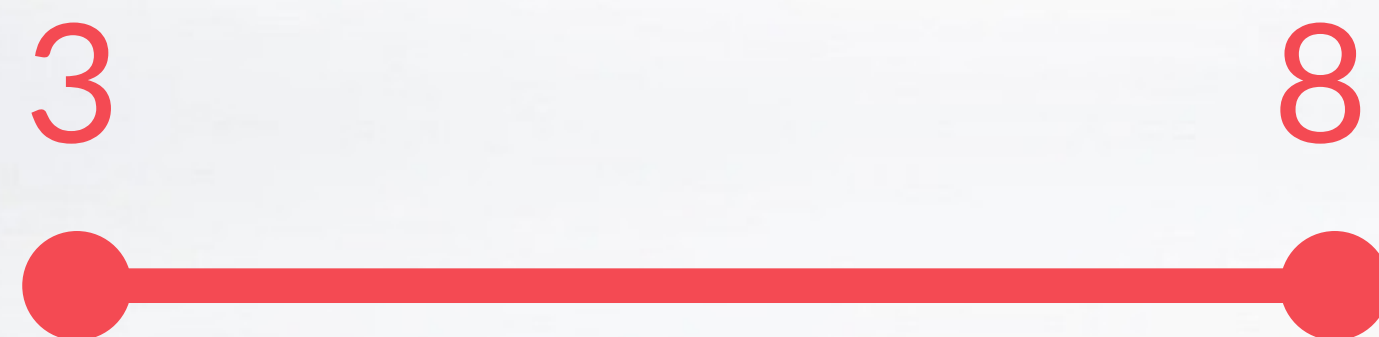


[再問] 用Math.ceil() 會發生什麼事？

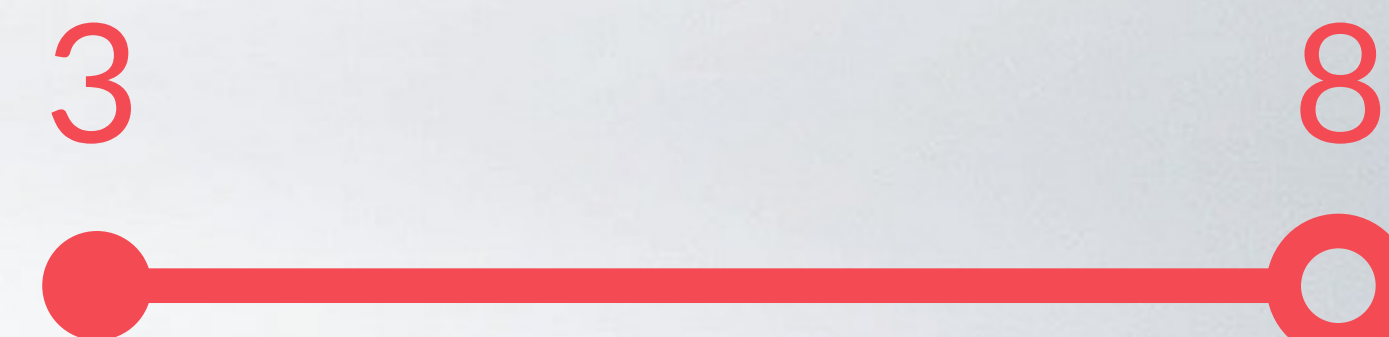
$$N \leq ? \leq M$$



$$3 \leq \text{Math.floor}(\text{Math.random()} * 5 + 3) \leq 7$$



$$3 \leq \text{Math.random()} * 5 + 3 < 8$$



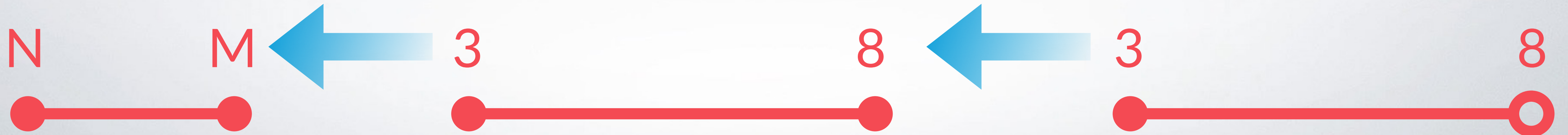


# 動腦解答

如何產生N至M之間的數？

$$3 \leq \text{Math.ceil}(\text{Math.random()} * 5 + 3) \leq 8$$

$$3 \leq \text{Math.random()} * 5 + 3 < 8$$



$$N \leq \text{Math.ceil}(\text{Math.random()} * (M - N) + N) \leq M$$



# JS 資料型態(type)

## 基本

布林 (Boolean)  
數值 (Number)  
字串 (String)

## 複合

陣列 (Array)  
函式 (Function)  
物件 (Object)



# 陣列

arr1 = [1,3,5]

arr2 = ["1", "3", "5"]

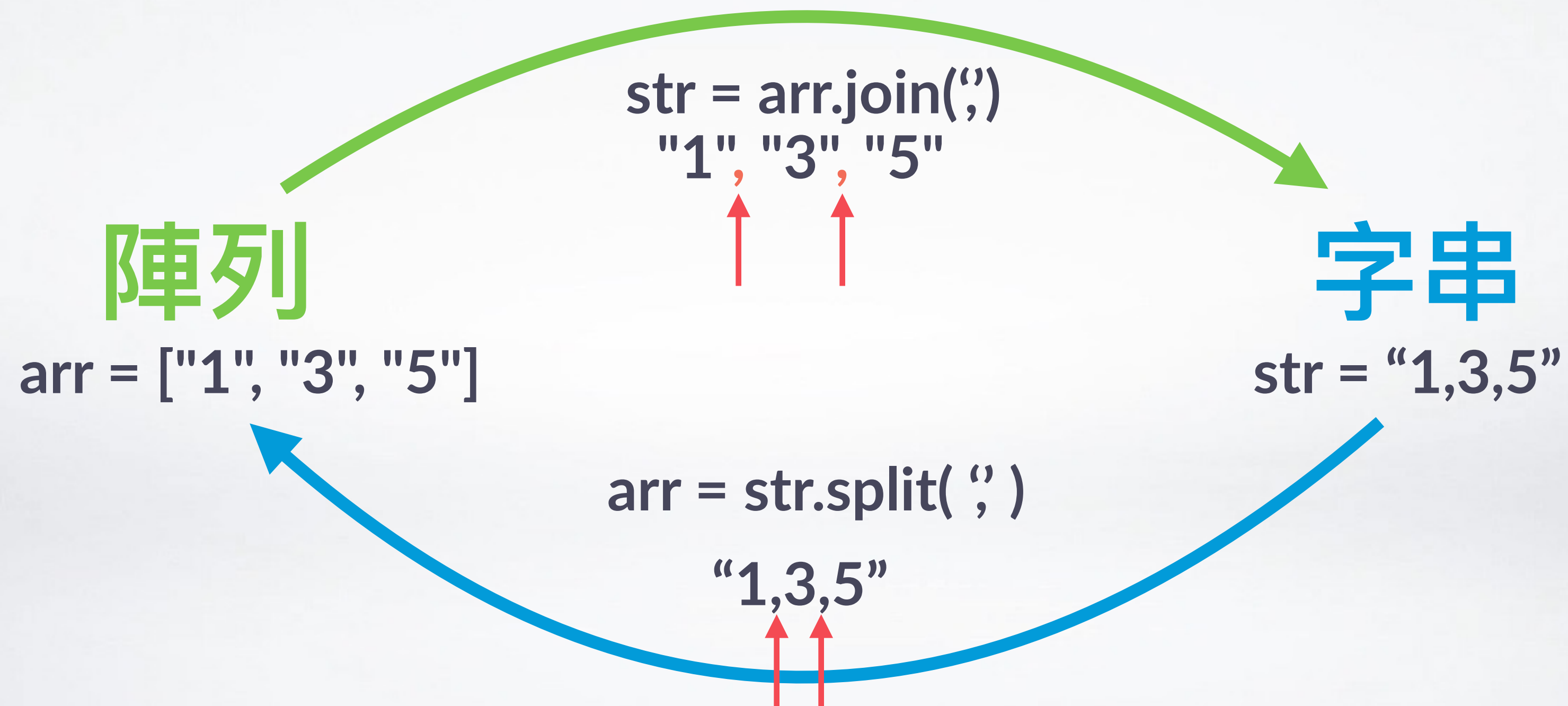
num	0	1	2
	1	3	5
str	"1"	"3"	"5"

# 陣列操作

功能	寫法	結果
陣列長度	arr1.length	3
取字元	arr2[1]	"3"
串接(前接後)	arr1.concat(arr2)	[1, 3, 5, "1", "3", "5"]
清空	arr1.length=0	arr1 內容為空
索引位置(元素)	arr2.indexOf('3')	1
	arr1.indexOf('3')	-1
陣列轉成字串([欲夾字串])	arr1.join()	"1,3,5"
	arr1.join(" and ")	"1 and 3 and 5"

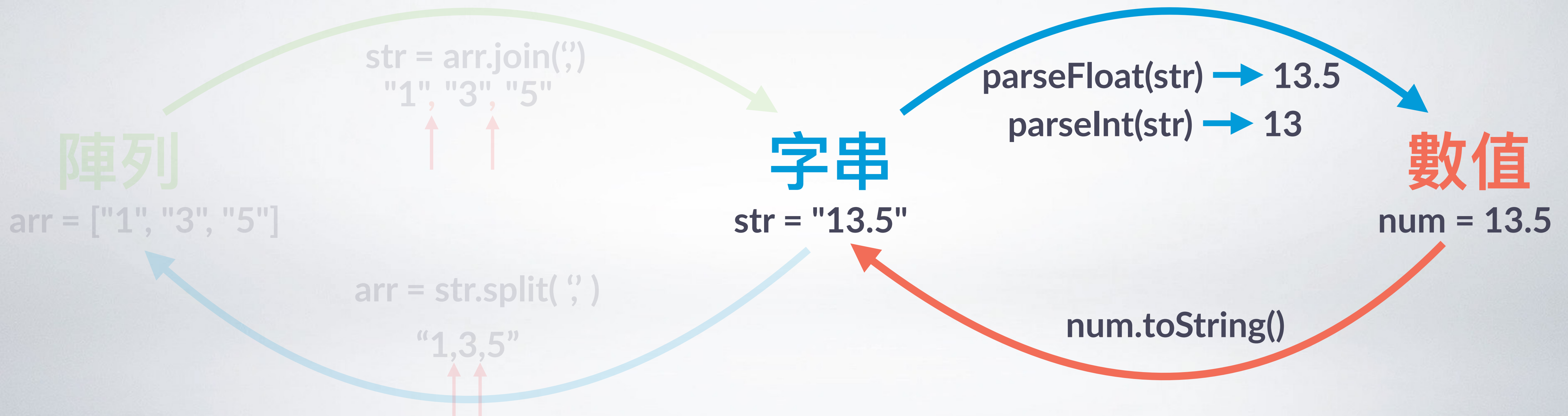


# 陣列 字串 互轉





# 陣列 字串 數值 互轉





# JS 資料型態(type)

## 基本

布林 (Boolean)  
數值 (Number)  
字串 (String)

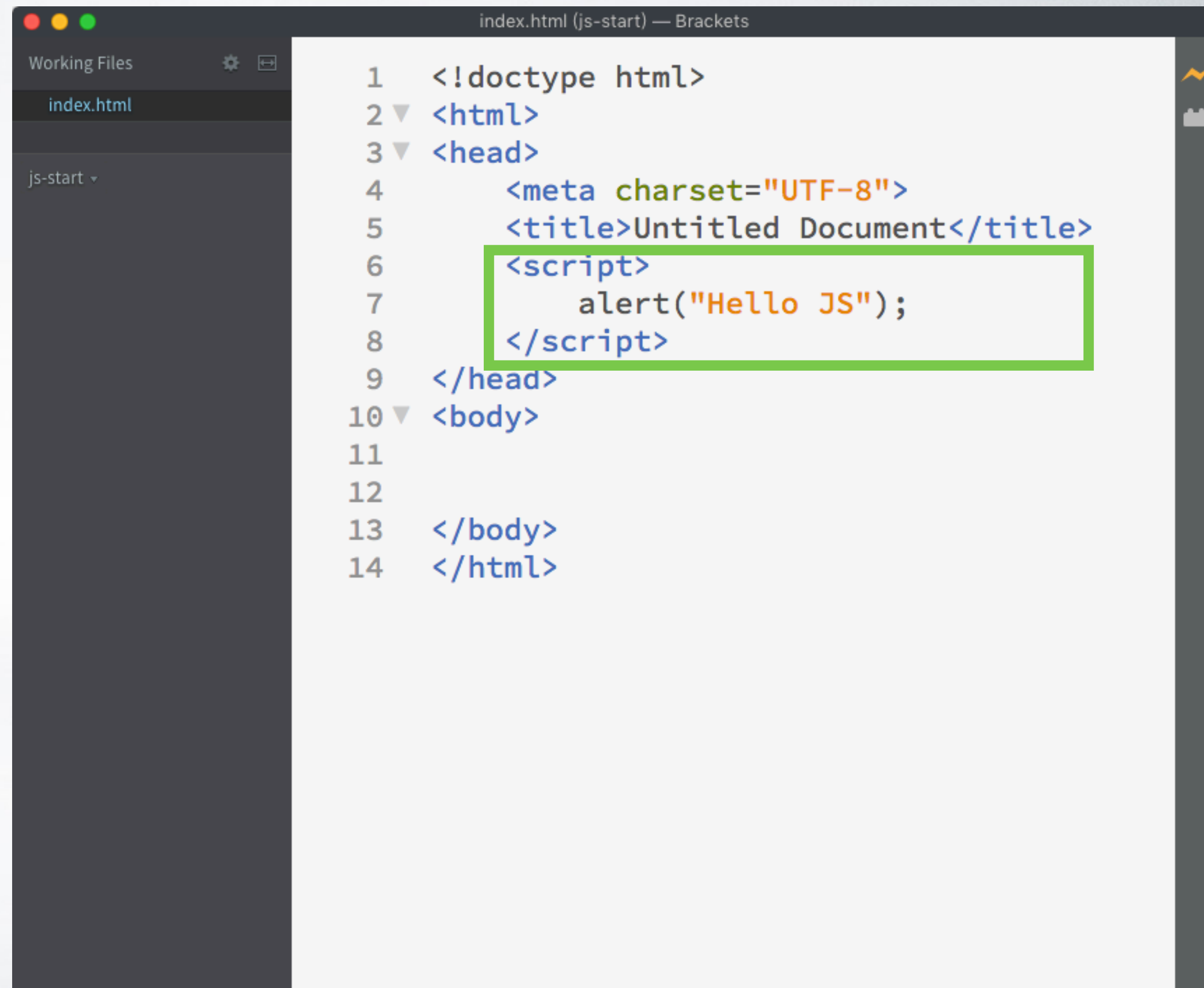
## 複合

陣列 (Array)  
函式 (Function)  
物件 (Object)



# JavaScript起手式

在<script>中寫 跳視窗:alert('Hello JS');



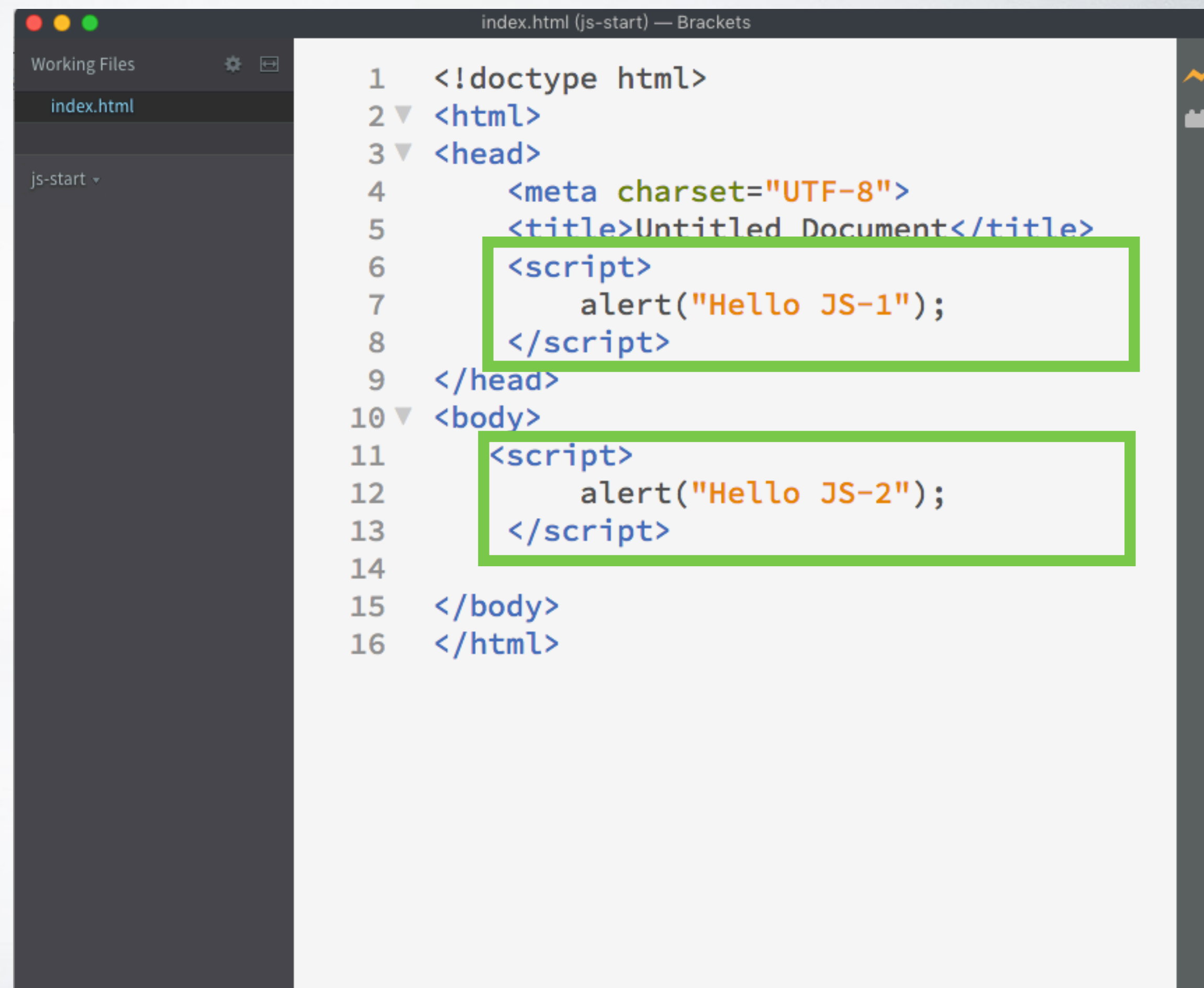
```
1  <!doctype html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Untitled Document</title>
6      <script>
7          alert("Hello JS");
8      </script>
9  </head>
10 <body>
11
12
13 </body>
14 </html>
```



# JavaScript起手式

小問答: 哪個alert()會先出現?

執行順序：從上往下，先head後body



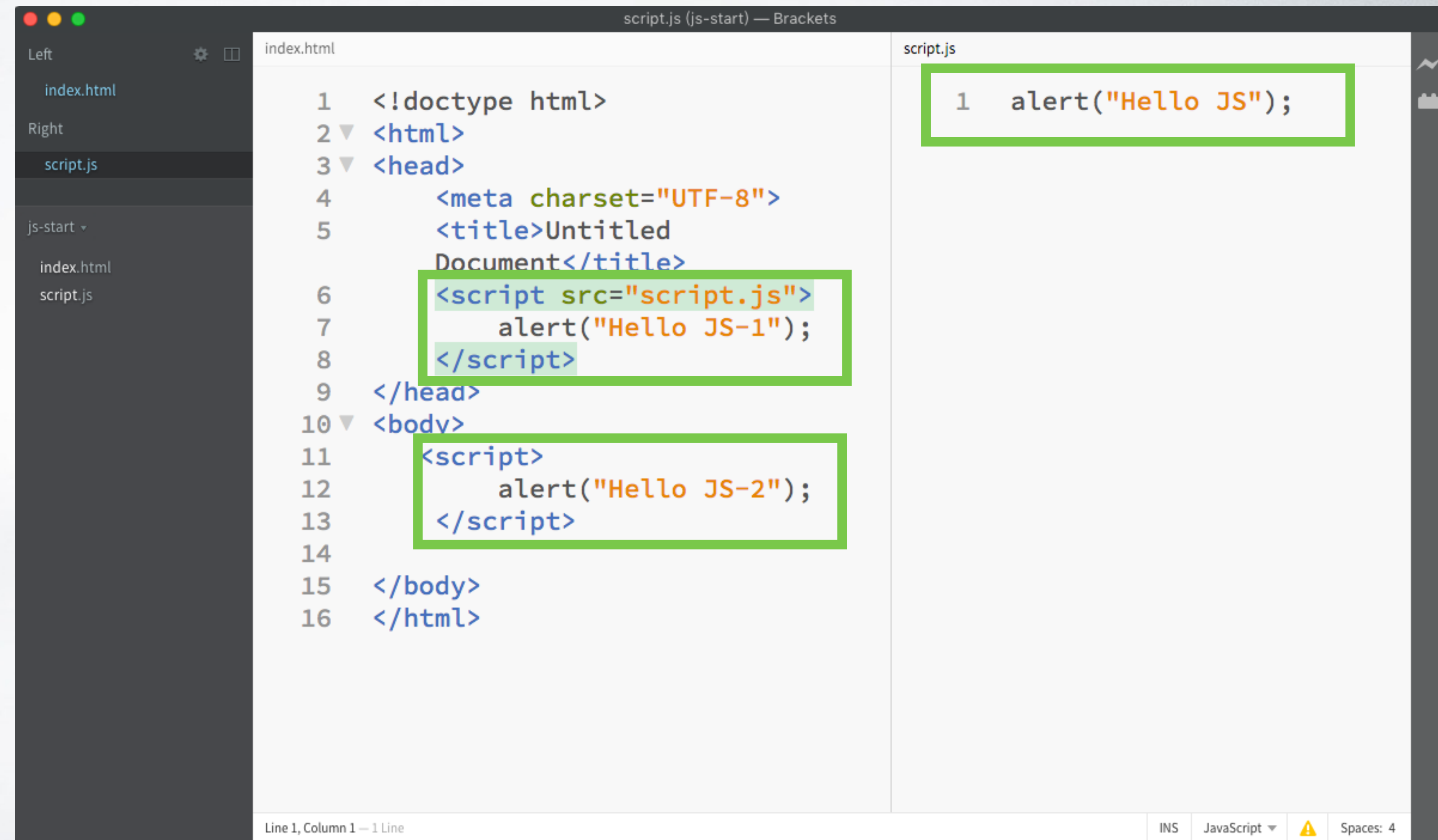
```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Untitled Document</title>
6   <script>
7     alert("Hello JS-1");
8   </script>
9 </head>
10 <body>
11   <script>
12     alert("Hello JS-2");
13   </script>
14
15 </body>
16 </html>
```



# 連結JavaScript

小問答: 哪個alert()會先出現?

執行順序：從上往下，先head後body



```
index.html
1  <!doctype html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Untitled Document</title>
6      <script src="script.js">
7          alert("Hello JS-1");
8      </script>
9  </head>
10 <body>
11     <script>
12         alert("Hello JS-2");
13     </script>
14
15 </body>
16 </html>
```

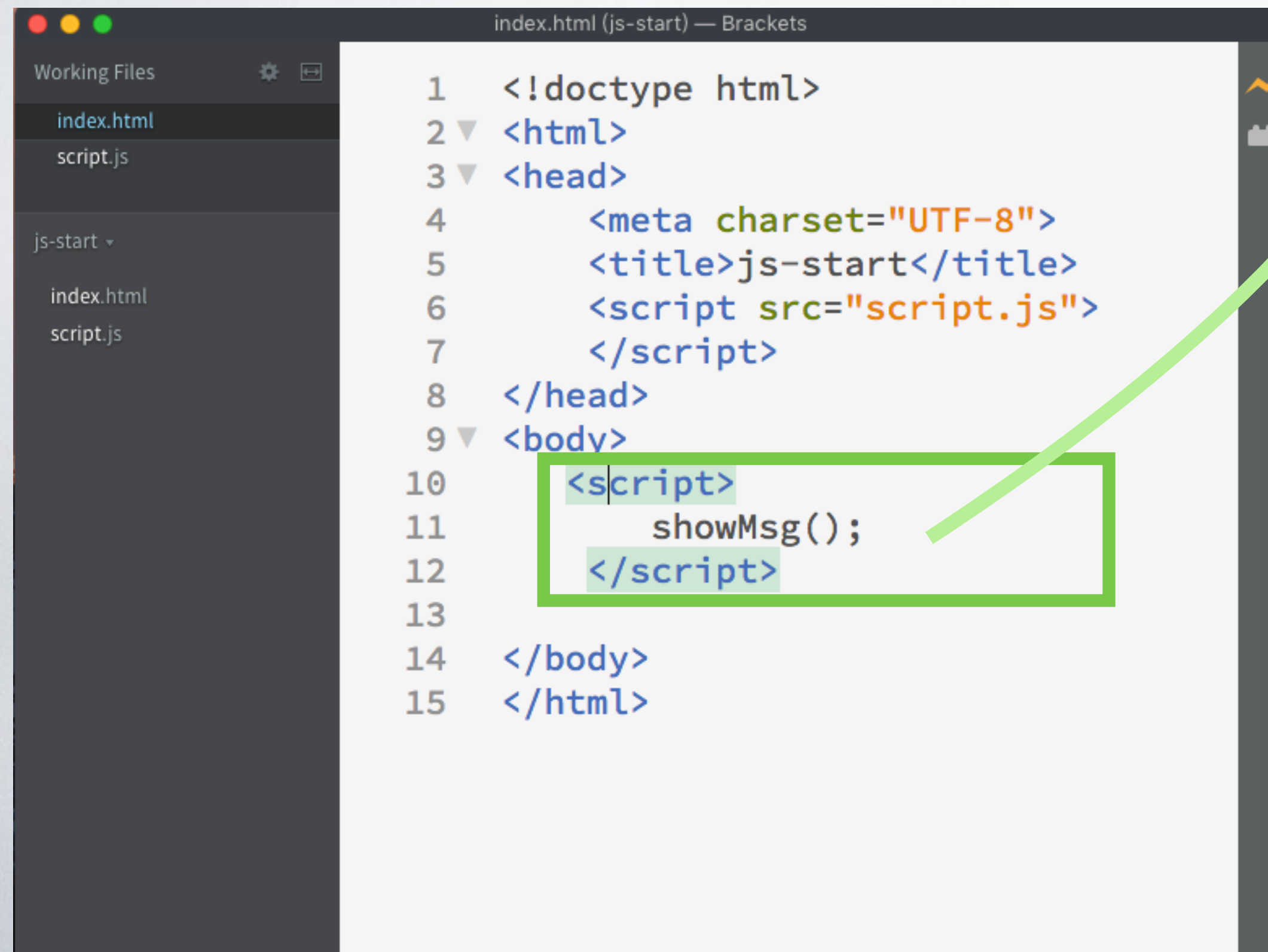
```
script.js
1  alert("Hello JS");
```



# 函式(Function)

把一段特定功能的程式碼打包起來的方法

## 最基本函式練習#1- 呼叫



The screenshot shows a code editor with a file named 'index.html (js-start) — Brackets'. The code is as follows:

```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>js-start</title>
6   <script src="script.js">
7   </script>
8 </head>
9 <body>
10  <script>
11    showMsg();
12  </script>
13
14 </body>
15 </html>
```

A green box highlights the script block in the body (lines 10-12), and a green arrow points from it to the function definition in the adjacent green box.

在 script.js 裡:

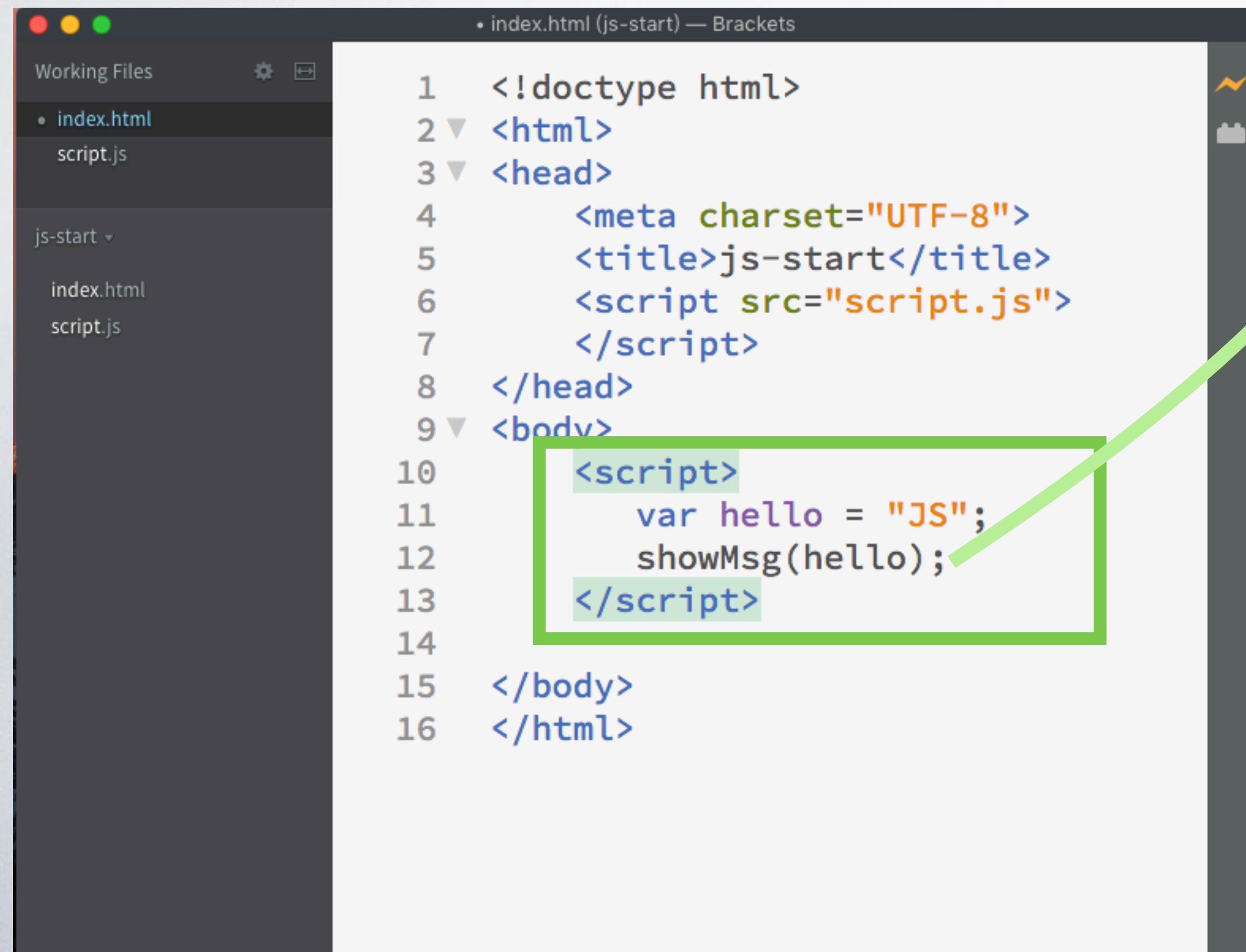
▼ 函式名稱(自訂)

```
function showMsg( ){
    alert("好好玩");
}
```



# 函式(Function)

## 最基本函式練習#2- 參數傳入



```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>js-start</title>
6   <script src="script.js">
7   </script>
8 </head>
9 <body>
10  <script>
11    var hello = "JS";
12    showMsg(hello);
13  </script>
14
15 </body>
16 </html>
```

在 script.js 裡:

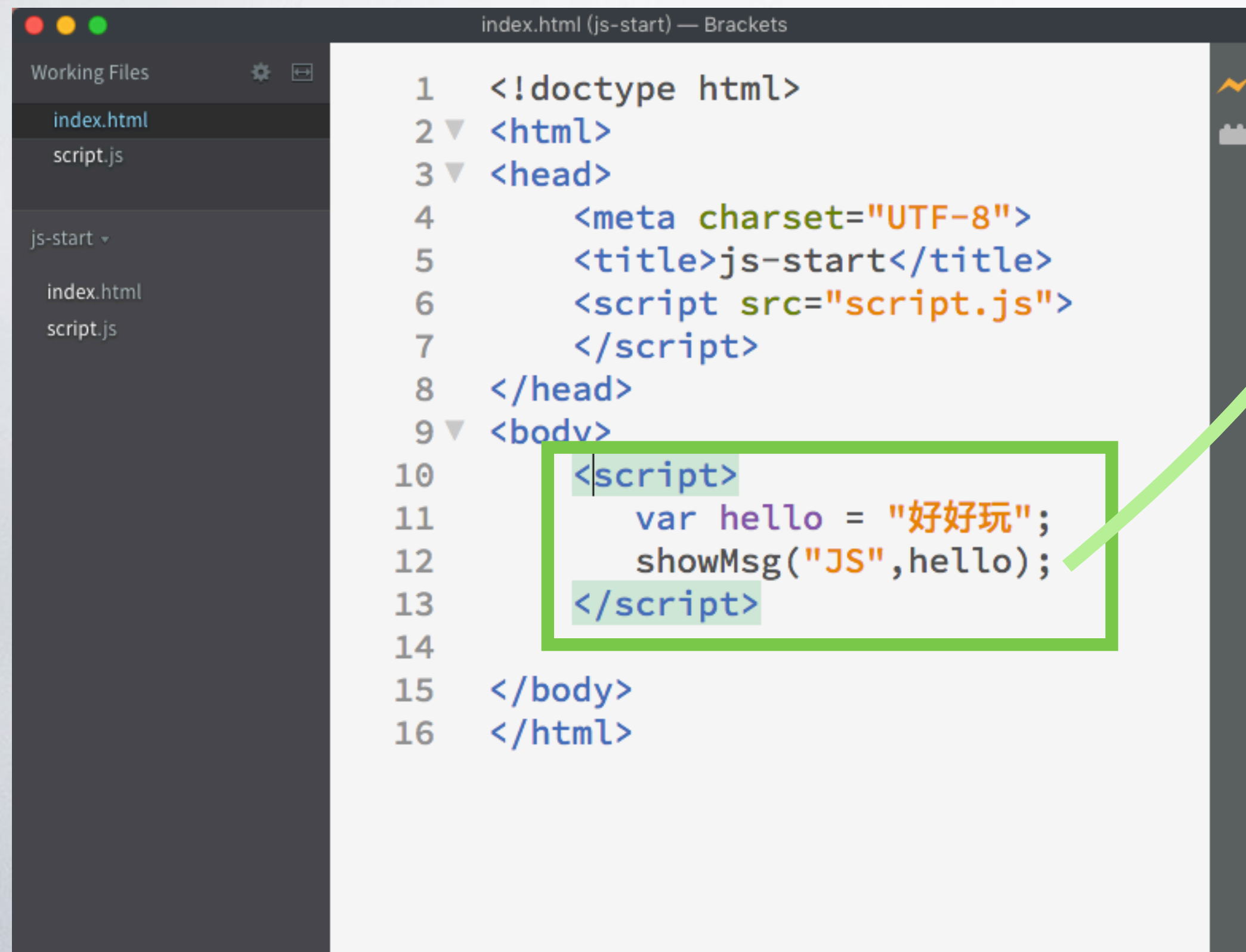
```
function showMsg(msg){
    alert(msg+"好好玩");
}
```

參數(作為內外溝通的變數)



# 函式(Function)

## 最基本函式練習#3- 多參數傳入



```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>js-start</title>
6   <script src="script.js">
7   </script>
8 </head>
9 <body>
10  <script>
11    var hello = "好好玩";
12    showMsg("JS",hello);
13  </script>
14
15 </body>
16 </html>
```

在 script.js 裡:

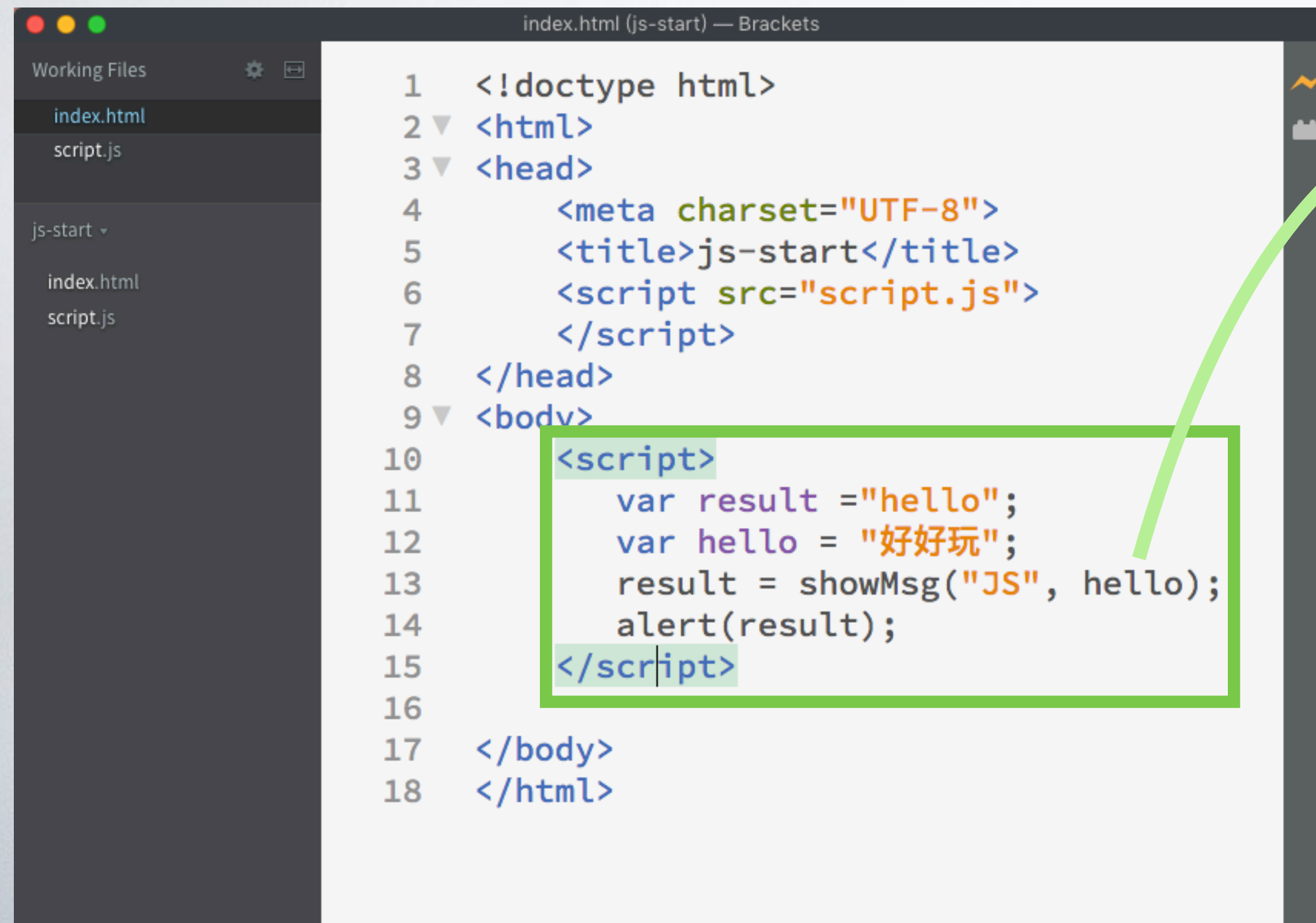
多個參數以逗號隔開

```
function showMsg(msg1, msg2){
  alert(msg1+msg2);
}
```



# 函式(Function)

## 最基本函式練習#4- 回傳



```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>js-start</title>
6   <script src="script.js">
7   </script>
8 </head>
9 <body>
10  <script>
11    var result = "hello";
12    var hello = "好好玩";
13    result = showMsg("JS", hello);
14    alert(result);
15  </script>
16
17 </body>
18 </html>
```

在 script.js 裡:

```
function showMsg(msg1, msg2){
  var msgAll = msg1+msg2;
  return msgAll;
  .....▶ 回傳(return)變數
  alert("我不會被執行");
}
```

試著用Chrome Console呼叫！





# 動手時間

做一個給他範圍:N,M  
的亂數產生的函式

用Chrome Console測試！

```
> random(1945,2016)
< 2013
> random(1945,2016)
< 2015
> random(1945,2016)
< 1980
> random(1945,2016)
< 2011
> random(1945,2016)
< 2000
> random(1945,2016)
< 1994
> random(1945,2016)
< 2012
> random(1945,2016)
< 1954
> random(1945,2016)
< 1997
> random(1945,2016)
< 2005
> |
```



# 答對了嗎

做一個給他範圍:N,M  
的亂數產生的函式

```
function random(n, m){  
    return Math.ceil(Math.random()*(m-n)+n);  
}
```



# 偵錯小函式 `console.log()`

從此不要再用`alert()`了

把`console.log(n,m)`  
放到`random`裡試試！

```
function random(n, m){  
    console.log(n, m);  
    return Math.ceil(Math.random()*(m-n)+n);  
}
```



# JS 資料型態(type)

## 基本

布林 (Boolean)  
數值 (Number)  
字串 (String)

## 複合

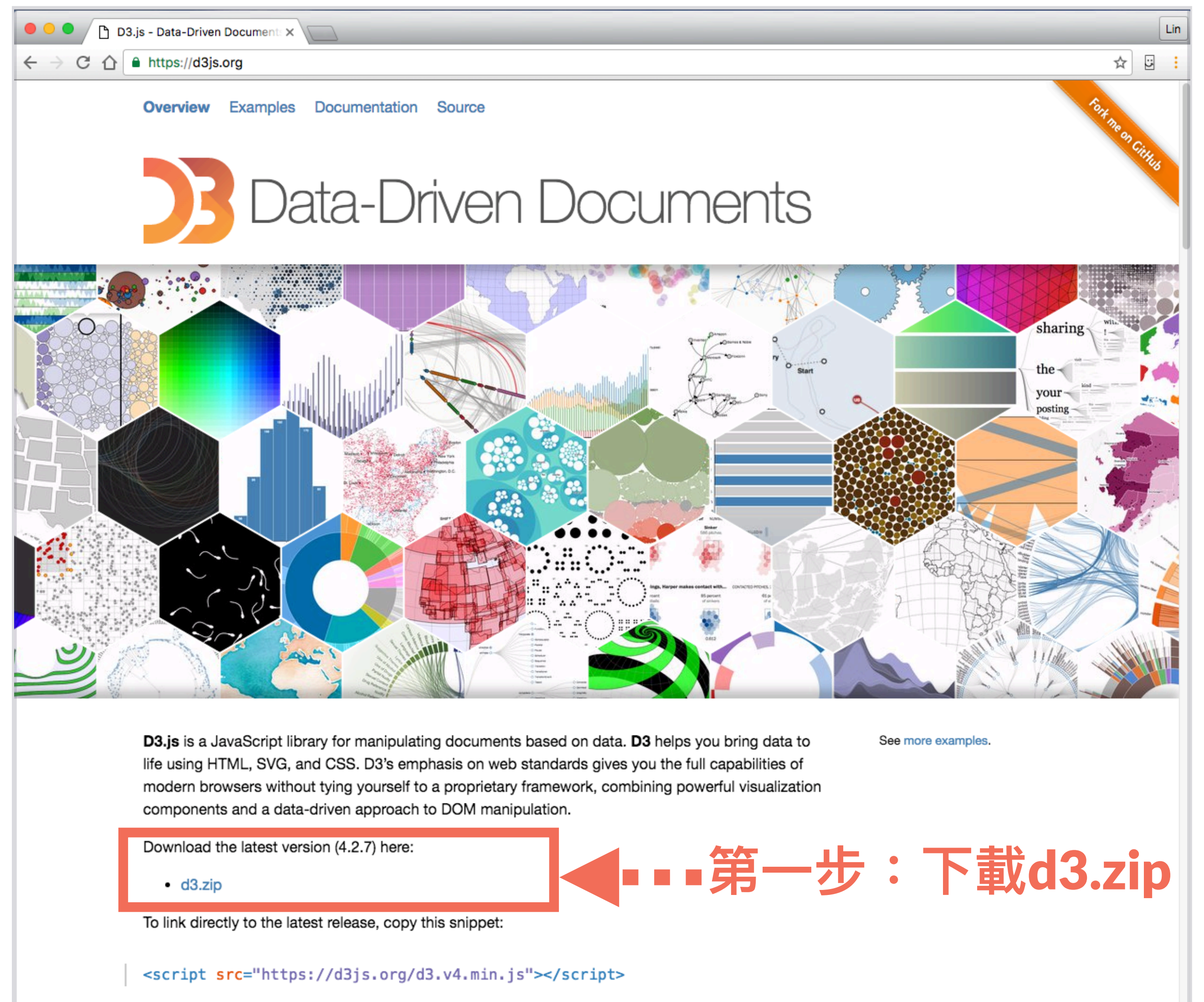
陣列 (Array)  
函式 (Function)  
物件 (Object)



# 如何開始使用D3.js

三步驟:

1. (到D3.js官網下載d3.zip)
2. 連結D3.js與你的HTML
3. 在<script>中寫D3/JS





# 使用D3第二步 - 加入D3.js函式庫到你的HTML中



The screenshot shows a code editor window titled "ex01.html (d3\_example) — Brackets". The code is as follows:

```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>Untitled Document</title>
6   <script src="https://d3js.org/d3.v3.min.js"></script>
7 </head>
8 <body>
```

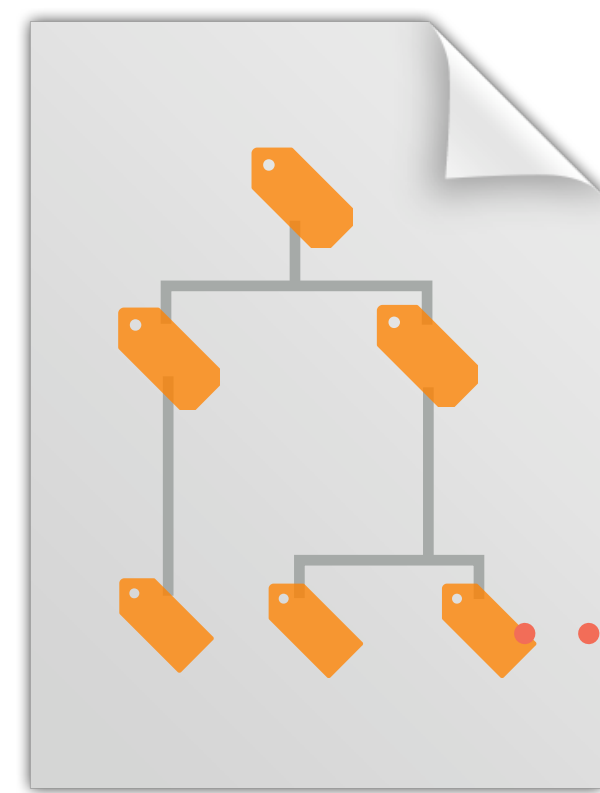
Two callout boxes provide instructions on how to include the D3.js library:

- 1. 從專案中 `<script src="d3.js"></script>`
- 2. 從網路上 `<script src="https://d3js.org/d3.v3.js"></script>`



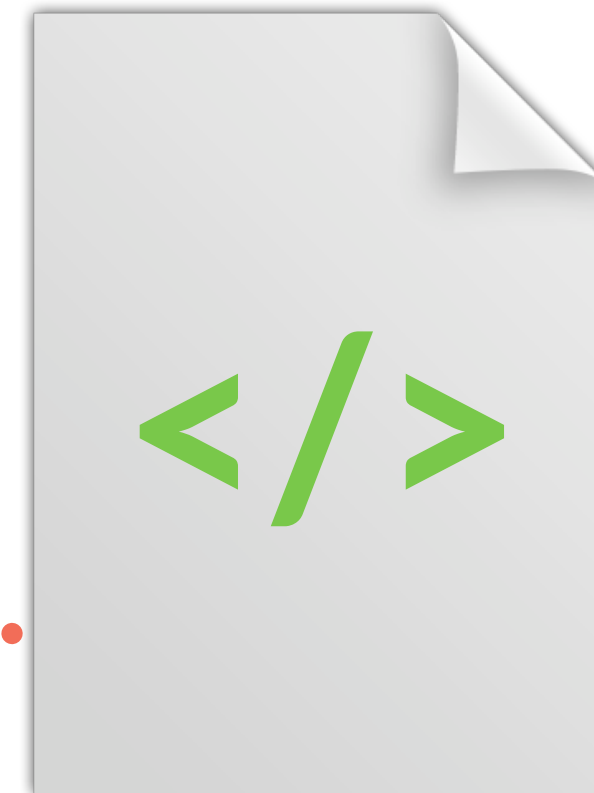
# 使用D3第二步 - 加入D3.js函式庫到你的HTML中

第二步: `<script src="https://d3js.org/d3.v3.js"></script>`



HTML

框架 / 內容



JavaScript

互動



## 第三步: 開始寫D3! - 在<script>裡寫D3/ JavaScript



The screenshot shows a code editor window titled "ex01.html (d3\_example) — Brackets". The left sidebar shows the "Working Files" panel with "ex01.html" selected. The main editor area displays the following HTML code:

```
1  <!doctype html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Untitled Document</title>
6      <script src="https://d3js.org/d3.v3.min.js"></script>
7  </head>
8  <body>
9      <script>
10         //在這裡寫D3
11     </script>
12 </body>
13 </html>
```

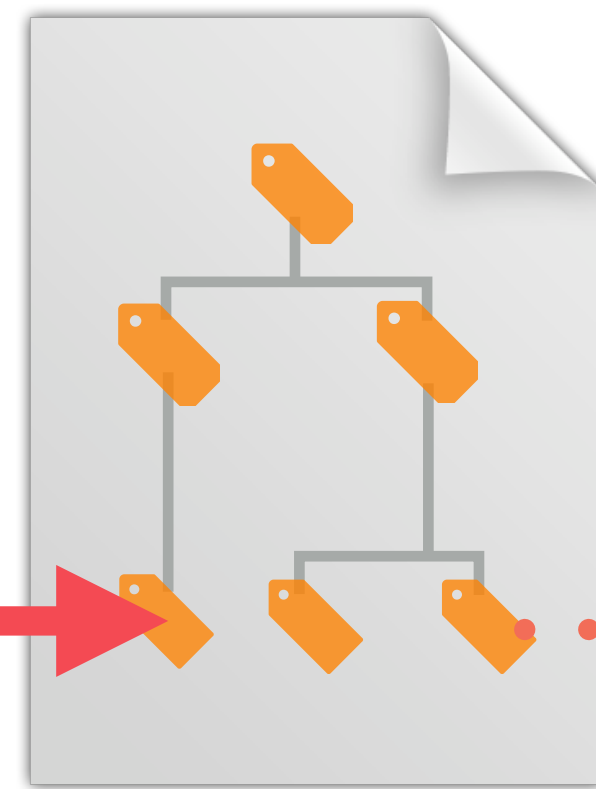
A red rectangular box highlights the script block on lines 9-11, which contains the opening and closing script tags and a comment indicating where to write D3.js code.



# 使用D3第三步 - 在<script>裡寫D3/ JavaScript

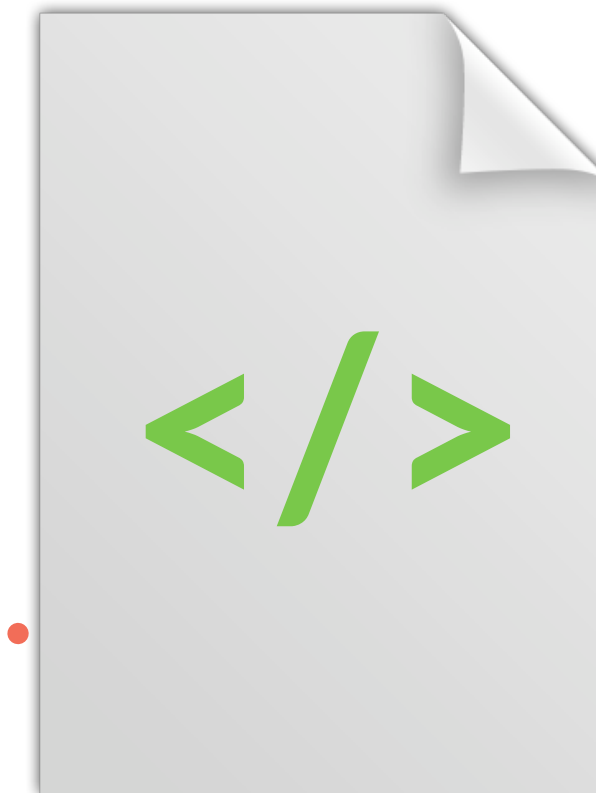
第二步: `<script src="https://d3js.org/d3.v3.js"></script>`

第三步:  
`<script> 在這裡寫D3 </script>`



HTML

框架 / 內容



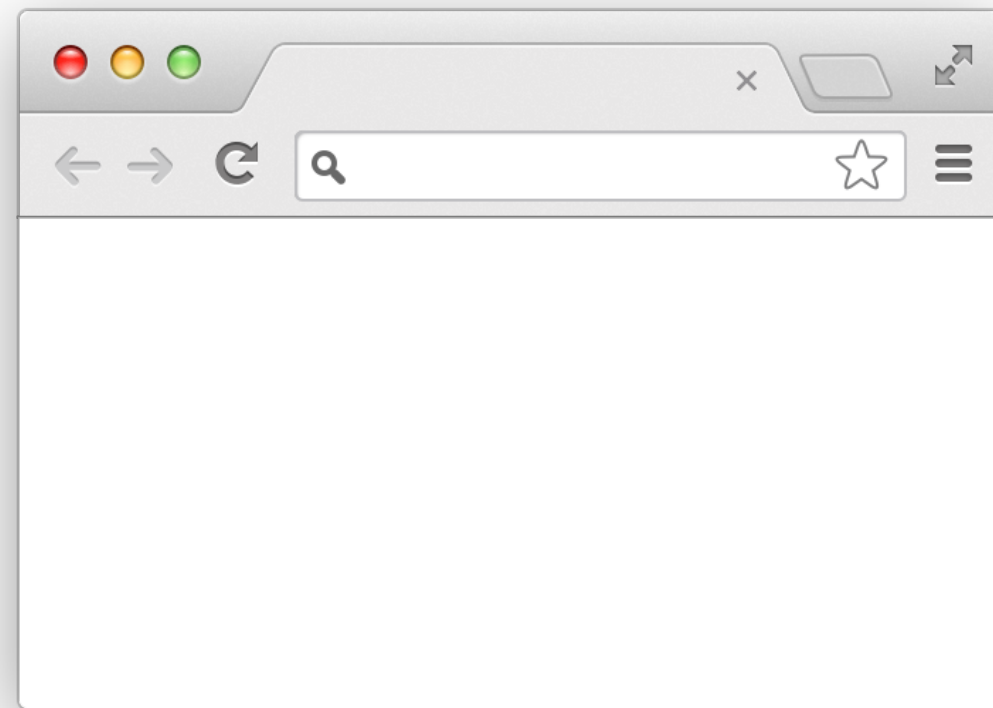
JavaScript

互動



# 先來認識 D3 語法

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>D3 Course</title>
  <script src="https://d3js.org/d3.v3.min.js"></script>
</head>
<body>
  <script>
    //在這裡寫D3
  </script>
</body>
</html>
```



# 鏈-結-語-法



# d3.select("body") ;

# 引用 D3

選擇器

行  
結  
束

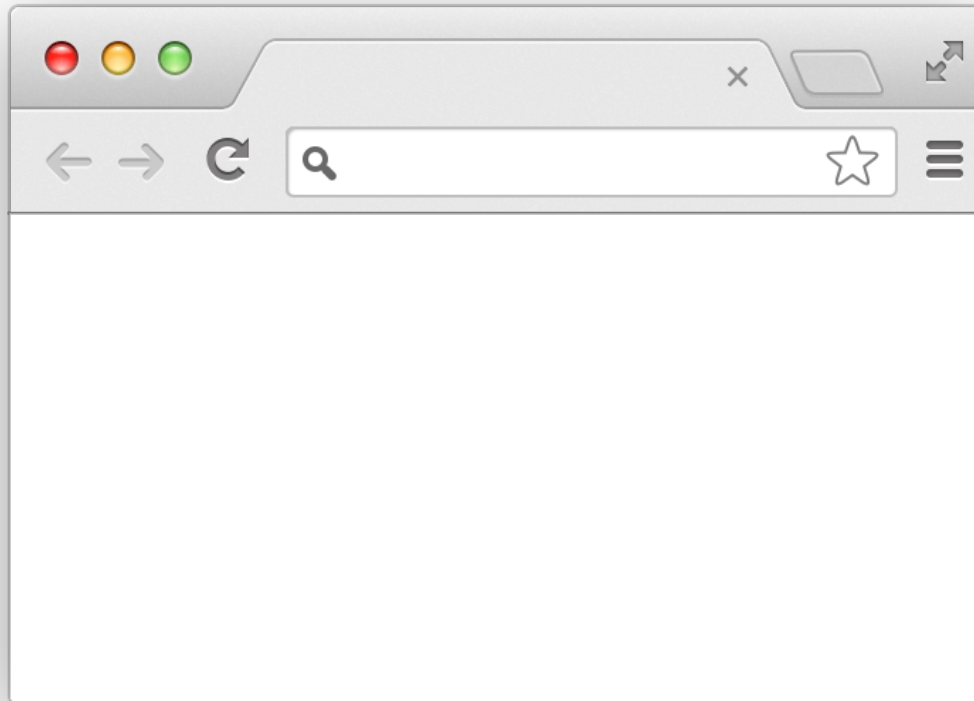
## 選第一個出現: .select()

選所有出現: `.selectAll()`



# 先來認識 D3語法

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>D3 Course</title>
  <script src="https://d3js.org/d3.v3.min.js"></script>
</head>
<body>
  <script>
    //在這裡寫D3
  </script>
  <div></div>
</body>
</html>
```



## 鏈-結-語-法



**d3.select("body").append("div");**

引  
用  
D3

選  
擇  
器

附  
加  
器

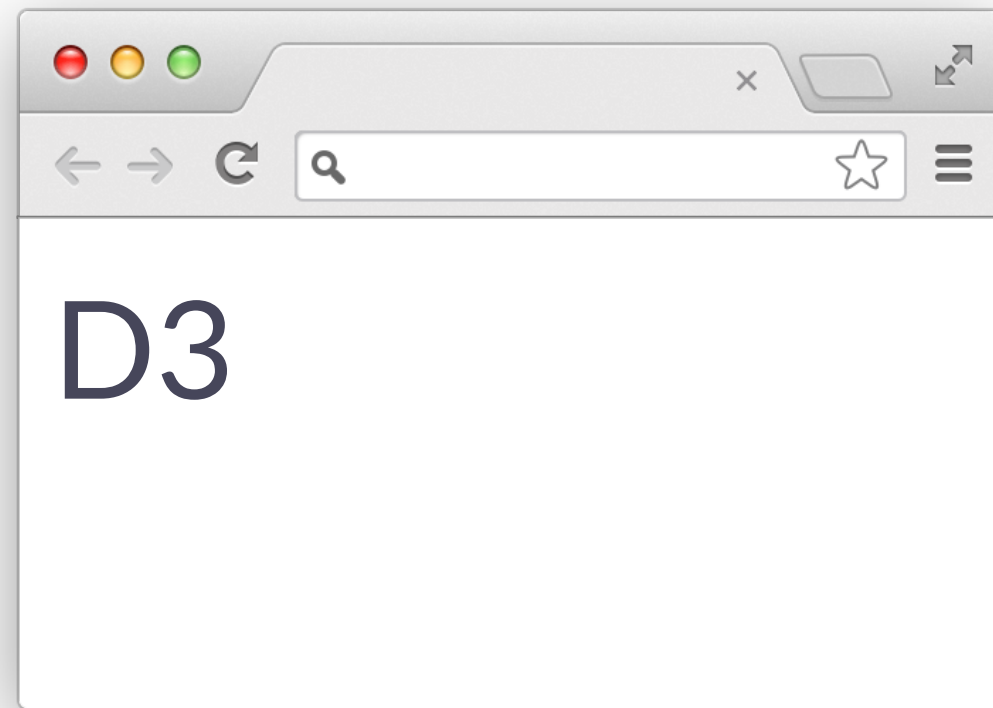
行  
結  
束

建立新元素，依附在最後



# 先來認識 D3語法

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>D3 Course</title>
  <script src="https://d3js.org/d3.v3.min.js"></script>
</head>
<body>
  <script>
    //在這裡寫D3
  </script>
  <div>D3</div>
</body>
</html>
```



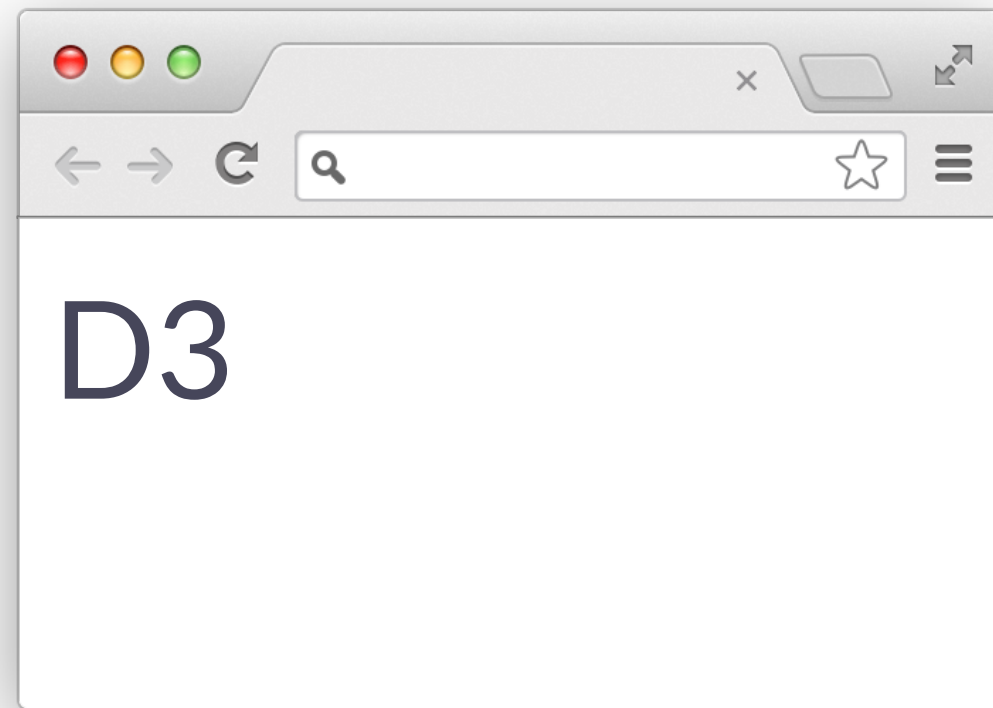
## 鏈-結-語-法





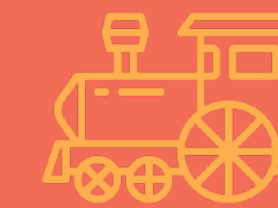
# 先來認識 D3語法

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>D3 Course</title>
  <script src="https://d3js.org/d3.v3.min.js"></script>
</head>
<body>
  <script>
    //在這裡寫D3
  </script>
  <div>D3</div>
</body>
</html>
```



## 鏈-結-語-法 (排版)

```
d3.select("body")
  .append("div")
  .text("D3");
```



Select

Append

Text

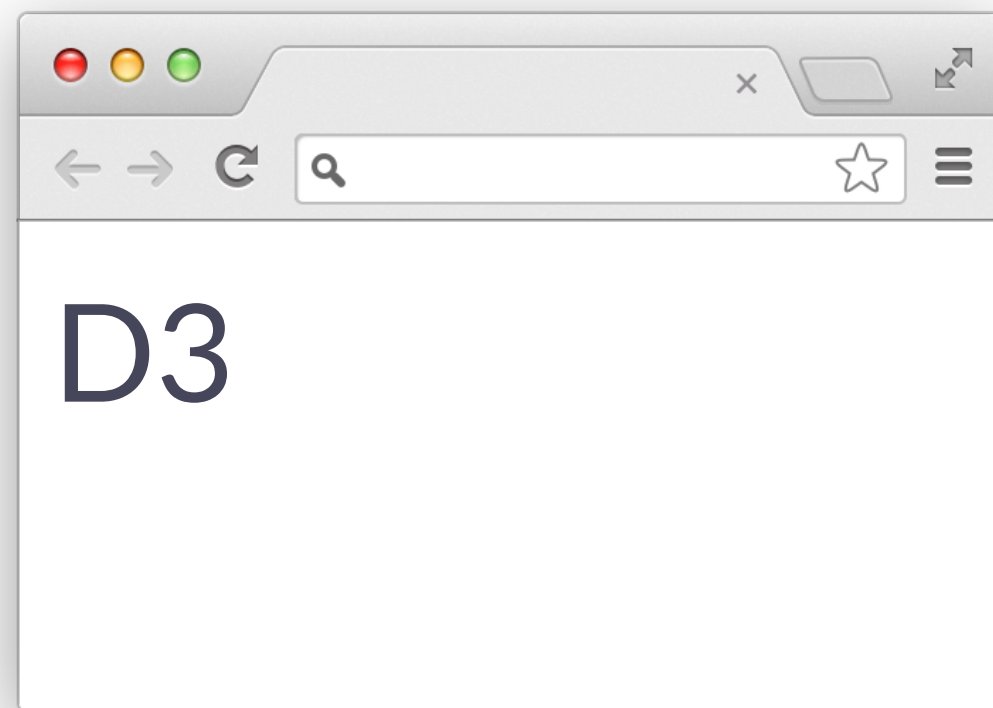


縮排請按[TAB]



# 先來認識 D3語法

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <title>D3 Course</title>
  <script src="https://d3js.org/d3.v3.min.js"></script>
</head>
<body>
  <script>
    //在這裡寫D3
  </script>
  <div>D3</div>
</body>
</html>
```



## 鏈-結-語-法 (存到變數裡)

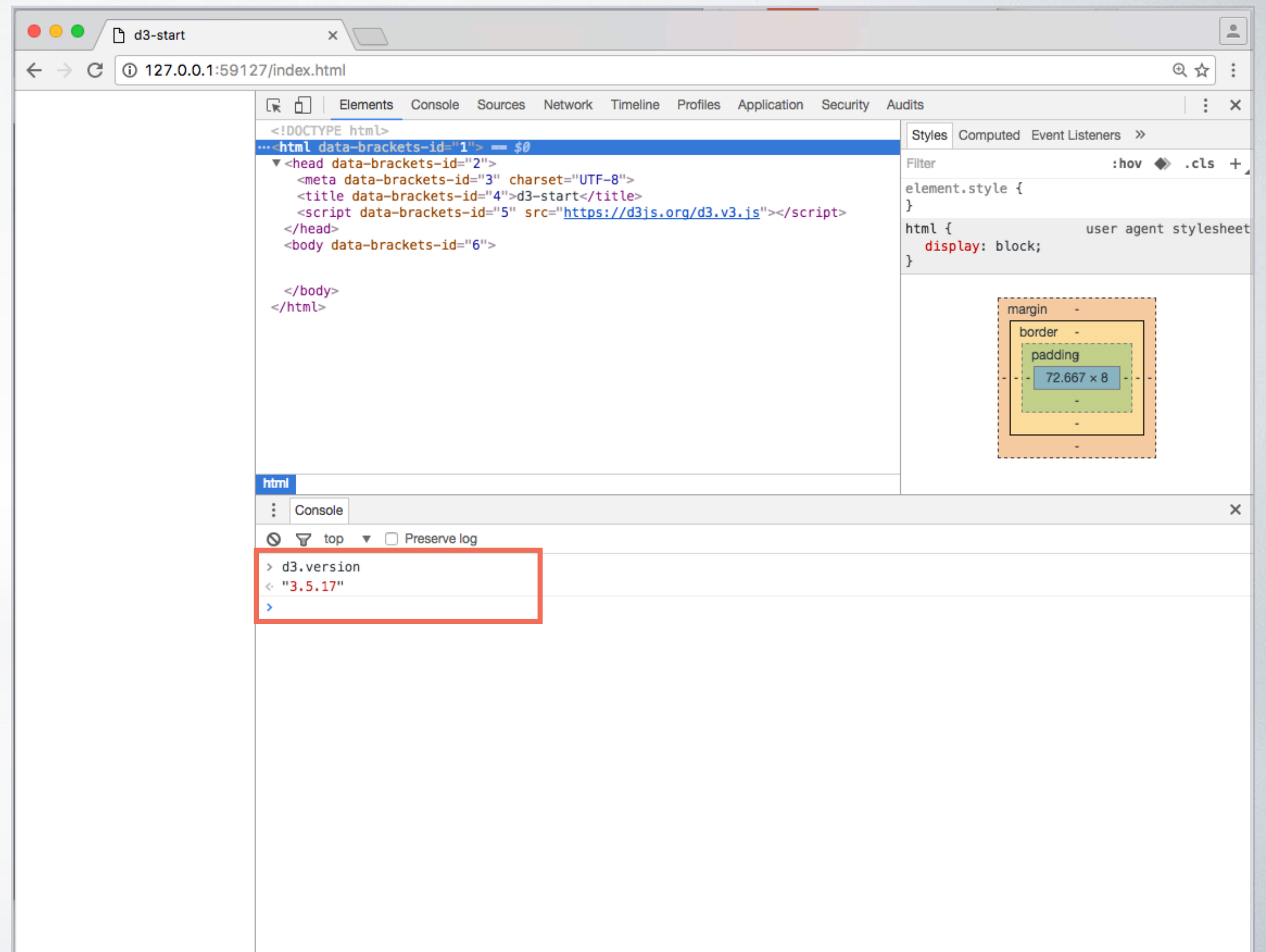
```
var body = d3.select("body");
```

```
body.append("div")
  .text("D3");
```



# 先試試console

1. 在console中先鍵入: d3.version
2. 在<body>中，加入有文字內容的<div>
3. 練習把body>div中的內容文字:D3換掉





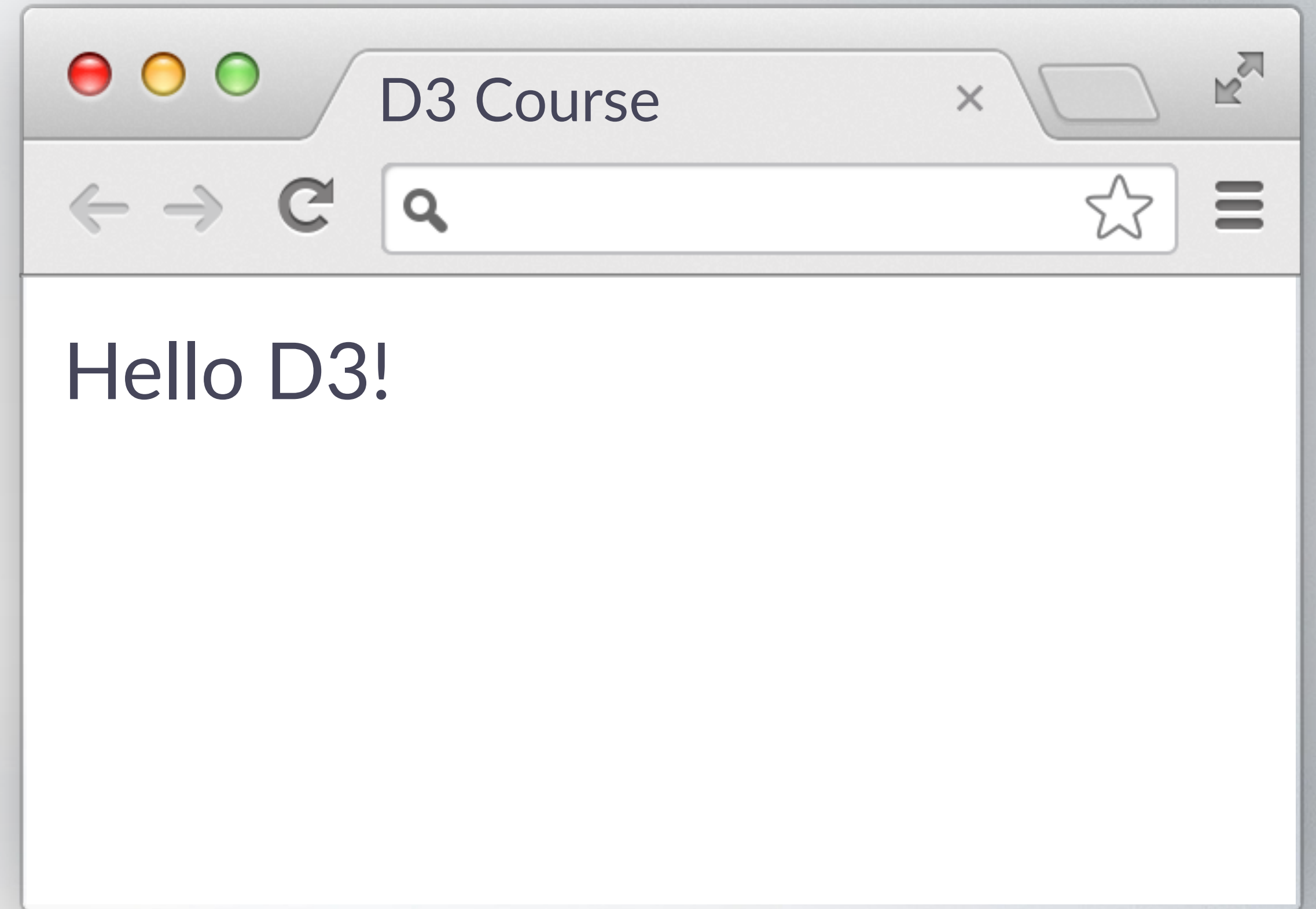
# 這兩個差在哪？

```
d3.select("body").append("div").text("D3");
```

```
d3.select("body").text("D3").append("div");
```

# 動手時間

在js腳本裡(不在console中)，  
使用**D3**語法做出來





# 一鍵即發

做一個啟動函式的按鈕  
按下後，會跳出一個介於  
1911~2016之間的數字

**按鈕:** `<input type="button" onclick="launch()" value="啟動">`

**函式:** `function launch(){ ... }`

`<p></p>`





# 解答在這裡

[按我前往](#)