



資料結構

陣列 - ARRAY

使用中括號群組多個數值，例如

- `a = [20, 10.3, "hi", [1, 2, 3]]`

陣列索引值

- 存取陣列時使用
- 第一筆資料的索引值為 0
- 例如 `a[2]` 為 "hi"

陣列越界

- 存取超過陣列索引值外的資料，例如 `x = a[10]`

陣列大小

`array.length`

例如：`n = [5, 3, 7].length`

列出陣列內容

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
for(i = 0; i < zoo.length; i++) {  
    console.log("動物園有" + zoo[i])  
}
```

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
for(i in zoo) {  
    console.log("動物園有" + zoo[i])  
}
```

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
for(animal of zoo) {  
    console.log("動物園有" + animal)  
}
```

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
zoo.forEach(animal =>  
    console.log("動物園有" + animal)  
)
```

新增與移除

unshift



push



["獅子", "老虎", "大象", "長頸鹿"]

shift



pop



常用函數1

是否包含

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
if (zoo.includes("大象")) {  
  console.log("動物園有大象")  
}
```

位於何處

```
zoo = ["獅子", "老虎", "大象", "長頸鹿"]  
index = zoo.indexOf("老虎")  
console.log(index) // 1
```

陣列反向

- arr.reverse()

```
zoo = ["獅子", "老虎", "大象", "長頸鹿", "老虎"]  
index = zoo.indexOf("老虎")  
console.log(index) // 1  
index = zoo.indexOf("老虎", index + 1)  
console.log(index) // 4
```

常用函數2

排序 (一律以字串的 unicode 排)

- `arr = [10, 5, 2, 9]`
- `arr.sort(function(a, b) {
 return a - b
})`
- `// arr == [2, 5, 9, 10]`

中文按筆畫數排

- `arr = ["二", "一", "嫁", "王"]
arr.sort(function(a, b) {
 return a.localeCompare(b, "zh-Hant")
})`
- `// arr == ['一', '二', '王', '嫁']`

使用數字陣列

- `arr = new Int32Array([10, 5, 2, 9])`
- `arr.sort()`
- `// arr == [2, 5, 9, 10]`

常用函數3

若 `arr = [0, 1, 2, 3]`

刪除元素內容 (陣列會留下一個「洞」)

- `delete arr[index]`

複製元素 `slice(start, end)`

- `new_arr = arr.slice(0, 2)`
- `// new_arr == [0, 1]`
- `new_arr = arr.slice(-2)`
- `// new_arr == [2, 3]`

刪除並插入 `splice(start, count)` 、 `splice(start, count, newEl)`

- `del_arr = arr.splice(1, 2, 5)`
- `// del_arr == [1, 2]`
- `// arr == [0, 5, 3]`

常用函數4

`map()`：將陣列中的每一個元素內容對映成另一內容

```
let a = [1, 2, 3, 4, 5]
let b = a.map(function(el) {
  return el * el
})
console.log(b) // b = [1, 4, 9, 16, 25]
```

練習

陣列洗牌

- 提示
- `Math.floor(Math.random() * 100)` 會產生 0 到 99 的隨機整數



物件 - OBJECT

OBJECT

物件中的屬性

```
var person = {  
  firstName: "David",  
  lastName: "Wang",  
  age: 30  
}  
  
console.log(person.firstName)  
// Prints "David"
```

物件中的方法

```
var person = {  
  firstName: "David",  
  lastName: "Wang",  
  age: 30,  
  name: function() {  
    return this.firstName + " " + this.lastName  
  }  
}  
  
console.log(person.name())  
// Prints "David Wang"
```

this 表示自己

SETTER 與 GATTER

```
var user = {  
  set name(newValue) {  
    this._name = newValue  
  },  
  get name() {  
    return this._name  
  }  
}
```

```
user.name = "David"  
console.log(user.name)
```

OBJECT 與 JSON

JSON 是一種輕量級的資料交換格式

例如

```
{  
  "month": "january",  
  "temp": 17  
}
```

```
[  
  {  
    "month": "january",  
    "temp": 17  
  },  
  {  
    "month": "february",  
    "temp": 15  
  }  
]
```

JSON 解析

將 JSON 字串轉成 Object 或 Array 型態

補充

```
// JSON Object 轉成 JSON 字串  
s = JSON.stringify(jsonObj);
```

注意 backtick 符號

```
var jsonString = `  
[  
  {  
    "month": "january",  
    "temp": 17  
  },  
  {  
    "month": "february",  
    "temp": 15  
  }  
]  
`;  
  
var jsonObj = JSON.parse(jsonString);  
for(data of jsonObj) {  
  console.log(data["month"]);  
}
```



集合 SET

特性

元素不可重複，無順序性

```
let set = new Set([1, 3, 2, 4, 4])  
console.log(set)
```

常用函數

大小：size

新增元素：add(element)

刪除元素：delete(element)

```
let set = new Set([1, 3, 2, 4, 4])  
let isOk = set.delete(4)
```

是否包含：has(element)



MAP

MAP

Key-value 組合成的陣列

Key 可為任何型態

```
var a = new Map([  
  ["key1", 10],  
  ["key2", "filename"]  
])
```

存入與取出

存入

- `.set("new_key", value)`

取出

- `.get("some_key")`

大小

- `.size()`

刪除

- `.delete("some_key")`