

JS Notebook v0.1

- 1. JavaScript Essentials
 - 1.1. Where to place the JS Code?
 - 1.1.1. inline
 - 1.1.2. internal
 - 1.1.3. external
 - 1.1.4. type="module"
 - 1.1.5. defer
 - 1.1.6. CDN
 - 1.2. 變量 Variables
 - 1.2.1. 為什麼需要變量？
 - 1.2.2. let & const
 - 1.3. 數據類型 Data Types
 - 1.3.1. number
 - 1.3.2. string
 - 1.3.3. boolean
 - 1.3.4. undefined
 - 1.3.5. null
 - 1.4. 類型轉換 Data Type Conversion
 - 1.4.1. number 轉 string
 - 1.4.2. string 轉 number
 - 1.5. 運算符 Operators
 - 1.6. 條件判斷 Conditions
 - 1.7. 循環 Loop
 - 1.7.1. for
 - 1.7.2. while
 - 1.7.3. break
 - 1.7.4. continue
 - 1.8. Function
 - 1.8.1. function declaration
 - 1.8.2. call the function
 - 1.8.3. local & outer variables
 - 1.8.4. parameters & default values

test.jsnb

Save

+

D

Comb

Split

List

unList

export

- 1.9.1. find elements
- 1.9.2. bind events

- 1.10. 練習

- 1.10.1. JS 計算器
- 1.10.2. Draggable Box
- 1.10.3. Strong Password Generator
- 1.11. 課外閱讀

1 JavaScript Essentials

1.1 Where to place the JS Code?

1.1.1 inline

不建議使用，但要知道有這種方式，課程後半段的 vue 會有大量類似的寫法，但和 inline 有著本質的不同

```
<a href="#" onclick="alert('hello'); return false;">Greeting</a>
<a href="javascript: alert('hello');">Greeting2</a>
```

1.1.2 internal

使用 `<script>` 包裹的 JS code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <script>
    alert('this is 1')
  </script>
</head>
<body>
  <script>
    alert('this is 2')
  </script>
</body>
</html>
```

1.1.3 external

獨立的 JS 檔案，使用 `<script src="/path/to/x.js">` 的方式引用

```
.
├─ index.html
```

```
└─ js
    └─ app.js
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <script type="module" src="/js/app.js"></script>
    <meta charset="UTF-8">
    <link rel="icon" href="/favicon.ico">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Just Notebook</title>
  </head>
  <body>
    <div id="app"></div>
    <script src="/js/app.js"></script>
  </body>
</html>
```

1.1.4 type="module"

1.1.5 defer

1.1.6 CDN

1.2 變量 Variables

1.2.1 為什麼需要變量？

+ = 11

body of index.html

```
<input type="number" id="a" />
<select id="operator">
  <option value="+">+</option>
  <option value="-">-</option>
  <option value="*">*</option>
  <option value="/">/</option>
</select>
<input type="number" id="b" />
<button></button>
<span id="result"></span>
```

internal script

```
function calc() {
  const inputA = document.querySelector("#a");
```

```
const inputB = document.querySelector("#b");
const operator = document.querySelector("#operator");
const result = document.querySelector("#result");

const a = inputA.value;
const b = inputB.value;
const op = operator.value;

if (op === "+") {
  result.innerText = a + b;
}

if (op === "-") {
  result.innerText = a - b;
}

if (op === "*") {
  result.innerText = a * b;
}

if (op === "/") {
  result.innerText = a / b;
}
}

const btn = document.querySelector("button");
// btn.onclick = calc;
btn.addEventListener("click", calc);
```

1.2.2 let & const

```
var a // 舊寫法，不建議使用

let a // 聲明一個會被改變的變量，此時 a 是 'undefined'
a = 1 // ok
a = 2 // ok
let a // Uncaught SyntaxError: Identifier 'a' has already been declared

// 「常量」
const a // Uncaught SyntaxError: Missing initializer in const declaration
const a = 1 // ok
a = 2 // not ok, Uncaught TypeError: Assignment to constant variable.
```

Scope

```
let a = 100;

function a() {
  console.log(a); // => 100
  let b = 200;
}
```

```
console.log(b); // => error
```

1.3 數據類型 Data Types

- number
- string
- boolean
- null
- undefined

1.3.1 number

```
1
1.0
0.1
.1
-0.1
1e-4 // 1/1000
2 ** 1024 // => Infinity
Number.MAX_SAFE_INTEGER // => 2 ** 53 - 1
```

1.3.2 string

使用引號括住，單引號、雙引號、反斜引號 3種符號都表示字串

```
'hello'
"hello"
'hello "John"'
"someone's package"
'someone\'s package'

`hello, I can use both 'a' and "b"`
```

1.3.3 boolean

```
true
false
```

1.3.4 undefined

```
let a // a => undefined
```

1.3.5 null

```
let a = null
```

1.4 類型轉換 Data Type Conversion

1.4.1 number 轉 string

```
String(100);  
(100).toString() // 要加 () 包住 number  
100 + ''
```

最常用的是 + "100"

1.4.2 string 轉 number

```
parseInt("100"); // => 100  
parseInt("100a") // => 100  
parseInt("a100") // => NaN
```

```
Number("100") // => 100  
Number("a100") // => NaN
```

```
parseInt("10101010", 2) // => 170
```

複雜字串中分析數值要用到正則表達式

```
"this is a string: 101.1".match(/[0-9.]+/);  
// => ['101.1', '101.1', index: 7, input: 'float: 101.1', groups: undefined]
```

1.5 運算符 Operators

```
1 + 2      // => 2  
1 - 2      // => -1  
1 * 2      // => 2  
1 / 2      // => 0.5  
2 ** 3     // => 8  
10 % 3     // => 1
```

簡寫

```
x = x + 1  
  
// 3 種簡寫  
// 1: x += 1  
// 2: x++
```

```
// 3: ++x
```

```
console.log(x++)      // => output 1, x = 2
console.log(++x)       // => output 3, x = 3
```

常用計算方法都可以簡寫：

```
x += 1
x -= 1
x *= 1
x /= 1
x %= 1
```

判斷運算符

```
1 == true      // => true
"1" == true    // => true
1 == "1"       // => true

1 === true     // => false
1 === "1"      // => false
1 === 1        // => true
```

不同數據類型的運算

```
1 + null       // => 1
1 + true       // => 2
1 + "1"        // => 11, not 2!
```

// 可以省去前面的數字，用 + 將其他數據類型轉為數字

```
+null         // => 0
+true         // => 1
+"1"          // => 1
```

```
"hello" + true      // "hellotrue"
"hello" + null       // "hellonull"
"hello" + undefined  // => "helloundefined"
```

// 用空字符將其他類型轉換為字符串

```
' ' + true      // => 'true'
' ' + null      // => 'null'
' ' + undefined // => 'undefined'
' ' + 123       // => '123'
```

// 一些特殊例子

```
1 / 0          // => Infinity
1 / "2"        // => 0.5
1 / "hello"    // => NaN
```

1.6 條件判斷 Conditions

```
if (a > 10) {  
    alert('a greater than 10')  
}
```

當執行語句只有一句，即 表達式 (expression) 時，可以省略 {}

```
if (a > 10) alert('a greater than 10')
```

```
if (a > 10) {  
    alert ('a greater than 10')  
} else if (a > 20) {  
    alert('a greater than 20')  
} else if (a > 30) {  
    alert('a greater than 30')  
} else {  
    alert('?')  
}
```

條件運算符

```
if (a > 10) {  
    alert('a greater than 10')  
} else {  
    alert('a less or equals to 10')  
}
```

```
const m1 = 'a greater than 10'  
const m2 = 'a less or equals to 10'  
const msg = a > 10 ? m1 : m2  
alert(msg)
```

```
a > 10 ? alert('a greater than 10') : alert('a less or equals to 10')
```

```
// or  
alert(a > 10 ? 'a greater than 10' : 'a less or equals to 10')
```

switch & case

```
switch(a) {  
    case 10  
        alert('a is 10')  
        break  
    case 20:
```



```
    alert('a is 20')
    break
  default
    alert('unknown')
}
```

1.7 循環 Loop

1.7.1 for

```
for(let i = 0; i < 100; i++) {
  console.log(i)
}
```

1.7.2 while

```
let i = 0
while (i < 10) {
  console.log("The number is " + i)
  i++
}
```

1.7.3 break

中斷並離開循環

1.7.4 continue

取消一步中後續的代碼，立即開始循環中的下一步

1.8 Function

1.8.1 function declaration

```
function hello() {
  alert('hello world')
}
```

function expression

```
const hello = function() {
  alert('hello')
}
```

1.8.2 call the function

```
hello()
```

1.8.3 local & outer variables

```
const greeting = 'good morning'

function sayHi(user) {
  alert(greeting + ' ' + user)
}
```

1.8.4 parameters & default values

```
function hello(userName = 'daniel') {
  alert('hello there! ' + userName)
}
```

1.8.5 return value

```
function add(a, b) {
  return +a + +b
}

const x1 = add(1, 2)           // => 3
const x2 = add("1", 2)        // => 3
const x3 = add(1, "2")        // => 3
const x4 = add("1", "2")      // => 3
```

1.8.6 callback

```
function greeting() {
  alert("hello there")
}

setTimeout(greeting, 1000)
```

1.9 DOM tree

1.9.1 find elements

```
document.querySelector('title')
document.querySelectorAll('a')
```

1.9.2 bind events

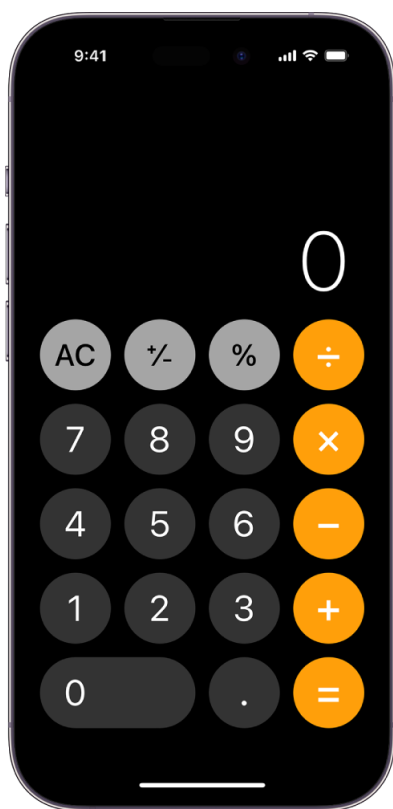
```
const el = document.querySelector('button')

function onBtnClick() {
  alert('btn been clicked!')
}

// el.onclick = onBtnClick
el.addEventListener('click', onBtnClick)
```

1.10 練習

1.10.1 JS 計算器





1.10.2 Draggable Box

1.10.3 Strong Password Generator

Strong Password Generator

Use the Strong Password Generator to create highly secure passwords that are difficult to crack or guess. Just select the criteria for the passwords you need and copy and paste.

nBZCz4.xZx"k*E(

Generate

Bookmark this page

Include Alpha Upper (A-Z): ☒

Include Alpha Lower (a-z): ☒

Include Number (0-9): ☒

Include Symbol: ☒

Length: 15

Transfer this password to your phone or tablet

Hold your device's camera to this QR code and your device will recognize the password encoded in the QR code.

On the destination device, [click here to go to the import password page](#)

This is the only way to assure you that at no point is the password transmitted over the internet and we have no record of it.



1.11 課外閱讀

<https://javascript.info/>

The Modern JavaScript Tutorial

<https://es6.ruanyifeng.com/>

ES6 入门教程

