Buổi 7
Tổng quan về JavaScript. Cấu trúc rẽ nhánh

I. Các khái niệm về JavaScript

JavaScript[1] là 1 ngôn ngữ kịch bản (*script*) cho phép lập trình viên cài đặt các hành động cho trang web phản hồi lại các thao tác từ người dùng.

Ví dụ:

- Khi người dùng nhấn vào 1 nút, một cửa sổ thông báo xuất hiện.
- Khi người dùng di chuyển chuột lên 1 tấm ảnh, tấm ảnh được phóng to.
- …

Có 3 cách để cài đặt JS cho 1 file HTML:

- External JS: Mã nguồn JS được lưu trong 1 file có đuôi *.js* và được chèn vào trang web thông qua thẻ `<script>` nằm trong thẻ `<head>` hoặc thẻ `<body>`:

```
<script src="js/script.js">
```

- Internal JS: Mã nguồn JS được đặt trong thẻ `<script>` nằm trong thẻ `<head>` hoặc thẻ `<body>`:

```
<script>
    // Mã nguồn JS
</script>
```

- Inline JS: Mã nguồn JS được đặt trong các sự kiện (*event*) của thẻ HTML:

```
<button onclick="…">Đăng nhập</button>
```

JS không chỉ được dùng trong các trang web mà còn ở các ứng dụng khác, ví dụ như Node.js. Ngoài ra, một số loại CSDL như MongoDB hay CouchDB cũng sử dụng JS làm ngôn ngữ lập trình.

Lưu ý: JavaScript và Java là 2 ngôn ngữ khác nhau.

II. Các thành phần cơ bản của JavaScript

1. Tên và cách đặt tên

Khi lập trình, thường phải đặt tên cho các biến số, hằng số, kiểu dữ liệu, hàm… trong chương trình để định danh và phân biệt chúng.

Tên trong JS chỉ được phép sử dụng chữ cái, chữ số, dấu gạch dưới nhưng không được phép có khoảng trắng, không có dấu tiếng Việt và không bắt đầu bằng chữ số.

Khi viết chương trình, nên đặt tên có tính gợi nhớ, diễn tả được ý nghĩa của đối tượng có tên đó nhằm giúp quá trình đọc hiểu code dễ dàng hơn.

2. Từ khóa

Từ khóa (*keyword*) là các từ dành riêng của JS. Lập trình viên không được phép đặt tên trùng với các từ khóa.

---

[1] Viết tắt là JS

Một số từ khóa của JS:

```
break       case        continue    default     do          for
function    if          new         return      switch      while
```

3. Chú thích

Chú thích (*comment*) là những đoạn văn bản mang ý nghĩa giải thích, ghi chú. Khi biên dịch, trình biên dịch sẽ bỏ qua những đoạn chú thích này.

Có 2 cách viết chú thích trong JS:

- Viết chú thích vào sau dấu `//`, có tác dụng đến hết dòng hiện tại.
- Viết chú thích vào giữa 2 dấu `/*` và `*/`.

Phím tắt để chú thích 1 đoạn lệnh trong VS Code là *Ctrl + K,C*.

4. Kết thúc câu lệnh

Mỗi câu lệnh trong JS phải được kết thúc bằng 1 dấu chấm phẩy (`;`). Với những đoạn lệnh dài đã nằm trong cặp dấu ngoặc `{ }`, khi kết thúc đoạn lệnh đó không cần dấu `;` nhưng mỗi câu lệnh trong đoạn lệnh đó vẫn phải kết thúc bằng dấu `;`.

5. Biến và kiểu dữ liệu

Biến (*variable*) đại diện cho 1 vùng nhớ dùng để lưu trữ dữ liệu. Biến thường được dùng để chứa giá trị nhập vào từ bàn phím hoặc kết quả các lệnh, phép toán. Mỗi biến có 1 tên riêng và không được trùng nhau. Muốn sử dụng biến phải khai báo trước.

Khác với một số ngôn ngữ lập trình, JS không ràng buộc kiểu dữ liệu khi khai báo biến, do đó, 1 biến có thể lưu nhiều kiểu dữ liệu khác nhau tùy vào giá trị mà nó được gán.

3 kiểu dữ liệu thường gặp nhất trong JS là Number, String và Boolean.

Cú pháp khai báo biến:

```
var <tên biến>;
```

Ví dụ:

```
var x;                  // Tại đây x là undefined
x = 5;                  // Tại đây x thuộc kiểu Number
x = "Hello";            // Tại đây x thuộc kiểu String
x = 16 + 4.0;           // Tại đây x có giá trị 20
x = 16 + 4 + "JS";      // Tại đây x có giá trị là "20JS"[2]
x = 16 + "JS" + 4;      // Tại đây x có giá trị là "16JS4"
```

JS hỗ trợ hàm `isNaN()` để kiểm tra 1 giá trị có phải là số hay không, và hàm `Number()` để chuyển 1 kiểu dữ liệu thành dạng số.

III. Toán tử

Trong ngôn ngữ lập trình, toán tử (*operator*[3]) là các kí hiệu dùng để thực hiện các phép toán.

Ví dụ: Trong câu lệnh `c = a + b;` thì `a`, `b`, `c` là toán hạng (*operand*) còn `+` và `=` là các toán tử.

1. Toán tử gán

Toán tử gán (*assignment op.*) dùng để gán giá trị cho biến. Giá trị gán có thể là 1 hằng số hoặc 1 biểu thức. Có thể kết hợp phép gán và câu lệnh khai báo biến. Cú pháp của phép gán:

```
[tên biến] = [giá trị/biểu thức];
```

---

[2] Phép cộng (+) khi thực hiện trên String sẽ là phép nối chuỗi

[3] Trong khuôn khổ môn này, operator viết tắt là op.

Ví dụ:

```
var a = 5;
var b, c;
b = a + 2;
a = a + 1;
c = 7;
```

2. Toán tử số học

Toán tử số học (*arithmetic op.*) dùng để thực hiện các phép toán số:

Toán tử	Ý nghĩa	Ví dụ
+	Cộng	5 + 7 → 12
-	Trừ	8.5 – 6 → 2.5
*	Nhân	4 * 9 → 36
/	Chia lấy thương	7 / 2 → 3.5
%	Chia lấy phần dư (modulo)	7 % 2 → 1
**	Lũy thừa	2 ** 3 → 8

Toán tử số học chỉ sử dụng trên các kiểu số.

Khi thực hiện phép + trên chuỗi và số, JS sẽ chuyển số về dạng chuỗi và thực hiện nối chuỗi.

Kết hợp toán tử số học và toán tử gán, ta có các toán tử sau: `+= -= *= /= %= **=`

Câu lệnh `a += b` tương đương với câu lệnh `a = a + b`.

3. Toán tử so sánh

Toán tử so sánh (*comparison op.*) dùng để so sánh 2 giá trị với nhau và trả về đúng (`true`) hoặc sai (`false`):

Toán tử	Ý nghĩa	Ví dụ
==	Bằng giá trị	5 == 2 + 3 → true
!=	Khác	7 != 9 → true
>	Lớn hơn	4 > 8 → false
>=	Lớn hơn hoặc bằng	7 >= 3 + 4 → true
<	Nhỏ hơn	-5.2 < 9 → true
<=	Nhỏ hơn hoặc bằng	-6.9 <= -9.6 → false
===	Bằng giá trị và giống kiểu dữ liệu	
!==	Khác giá trị hoặc khác kiểu dữ liệu	

Phân biệt toán tử == và ===, != và !==: Giả sử `x = 5`:

Toán tử	Ý nghĩa	Biểu thức	Giá trị
==	bằng giá trị	x == 8 x == 5 x == "5"	false true true
===	bằng giá trị **và** cùng kiểu	x === 5 x === "5"	true false
!=	khác giá trị	x != 8	true
!==	khác giá trị **hoặc** khác kiểu	x !== 8 x !== 5 x !== "5"	true false true

Khi thực hiện phép so sánh trên chuỗi và số, JS sẽ chuyển chuỗi về dạng số trước khi tiến hành so sánh. Nếu chuỗi không thể chuyển về dạng số thì xem như giá trị là `NaN`[4] và kết quả phép so sánh luôn là `false`:

Biểu thức	Giá trị
`2 < 12`	`true`
`2 < "12"`	`true`
`2 < "John"`	`false`
`2 > "John"`	`false`
`2 == "John"`	`false`
`"2" < "12"`	`false`
`"2" > "12"`	`true`
`"2" == "12"`	`false`

4. Toán tử logic

Toán tử logic[5] (*logical op.*) dùng để nối các mệnh đề logic và cho kết quả là `true` hoặc `false`:

Toán tử	Ý nghĩa	Ví dụ
`!`	Phủ định	`!(5 < 7)` → `false`
`&&`	Và	`(8.5 > 16) && (7 != 2)` → `false`
`||`	Hoặc	`(4 > 8) || (2 == 2)` → `true`

5. Toán tử tăng, giảm

Toán tử tăng, giảm (*increment & decrement op.*) dùng để tăng, giảm giá trị của biến 1 đơn vị.

Câu lệnh `x++;` tương đương với câu lệnh `x = x + 1;`

Câu lệnh `x--;` tương đương với câu lệnh `x = x - 1;`

IV. Cấu trúc rẽ nhánh

Trong lập trình, khi giải quyết 1 bài toán, đôi lúc cần phải đưa ra lựa chọn tùy thuộc vào 1 điều kiện nào đó.

Ví dụ: Tính giá trị của hàm số f(x), biết rằng:

$$f(x) = \begin{cases} 7x + 2 \text{ nếu } x \geq 0 \\ x^2 - 6 \text{ nếu } x < 0 \end{cases}$$

Có thể thấy rằng trong ví dụ trên, tùy thuộc vào giá trị của x nhập vào mà cách tính giá trị hàm số f(x) có sự thay đổi.

Khi gặp những trường hợp cần đưa ra lựa chọn, chúng ta sử dụng cấu trúc rẽ nhánh.

1. Câu lệnh if

Câu lệnh `if` cho phép chương trình lựa chọn có thực hiện một công việc nào đó hay không dựa trên việc kiểm tra điều kiện nào đó là đúng hay sai. Điều kiện phải là một biểu thức, mệnh đề đúng hoặc sai. Nếu điều kiện đúng thì thực hiện công việc, nếu điều kiện sai thì bỏ qua.

Cú pháp:

```
if (<biểu thức điều kiện>) {
      // Các câu lệnh cần thực hiện nếu điều kiện đúng
}
```

[4] Not A Number

[5] Còn gọi là toán tử luận lý

Lưu ý:

- Tất cả điều kiện đặt trong cặp dấu ngoặc ( ).
- Khi có nhiều điều kiện, sử dụng các toán tử && và || để nối các điều kiện với nhau.
- Nếu chỉ có 1 câu lệnh cần thực hiện thì có thể bỏ cặp dấu ngoặc { }, tuy nhiên, **khuyến cáo** luôn dùng cặp dấu ngoặc { } bất kể có 1 hay nhiều câu lệnh.

Ví dụ: Kiểm tra xem biến x có phải là số nguyên dương hay không:

```
if (x >= 0) {
	alert("x là số nguyên dương");
}
```

Ví dụ: Kiểm tra xem biến a có phải là số nguyên dương chẵn hay không:

```
if (a % 2 == 0 && a >= 0) {
	alert("a là số nguyên dương chẵn");
}
```

2. Câu lệnh if-else

Câu lệnh `if-else` cho phép chương trình lựa chọn thực hiện công việc A hay công việc B dựa trên việc kiểm tra điều kiện nào đó là đúng hay sai. Điều kiện phải là một biểu thức, mệnh đề đúng hoặc sai. Nếu điều kiện đúng thì thực hiện công việc A, nếu điều kiện sai thì thực hiện công việc B.

Cú pháp:

```
if (<biểu thức điều kiện>) {
	// Các câu lệnh cần thực hiện nếu điều kiện đúng
}
else {
	// Các câu lệnh cần thực hiện nếu điều kiện sai
}
```

Ví dụ: Kiểm tra xem biến x là số nguyên dương hay số âm:

```
if (x >= 0) {
	alert("x là số nguyên dương");
}
else {
	alert("x là số âm");
}
```

Ví dụ: Kiểm tra xem biến a có phải là số nguyên dương chẵn hay không:

```
if (a % 2 == 0 && a >= 0) {
	alert("a là số nguyên dương chẵn");
}
else {
	alert("a không phải là số nguyên dương chẵn");
}
```

3. Câu lệnh switch

Câu lệnh `switch` cho phép chương trình lựa chọn thực hiện 1 trong số nhiều công việc tùy thuộc vào giá trị của 1 biểu thức.

Cú pháp:

```
switch (<biểu thức>) {
	case <giá trị 1>: <công việc 1>; break;
	case <giá trị 2>: <công việc 2>; break;
```

```
        case <giá trị 3>: <công việc 3>; break;
        …
        default: <công việc 0>;
    }
```

Trong cú pháp trên, chương trình sẽ tính toán giá trị biểu thức, nếu kết quả là giá trị 1 thì thực hiện công việc 1, nếu kết quả là giá trị 2 thì thực hiện công việc 2… Nếu không có kết quả nào trùng khớp thì thực hiện công việc 0.

Lưu ý:

- Biểu thức đặt trong cặp dấu ngoặc `( )`.
- Toàn bộ phần thân của câu lệnh `switch` phải đặt trong cặp dấu ngoặc `{ }` cho dù chỉ có 1 mệnh đề `case` duy nhất.
- Giá trị của biểu thức và giá trị ở các mệnh đề `case` phải cùng kiểu dữ liệu.
- Tất cả các giá trị trong các mệnh đề case phải khác nhau, và mỗi mệnh đề `case` chỉ chứa 1 giá trị.
- Mệnh đề `default` không bắt buộc.
- Câu lệnh `break` dùng để thoát khỏi câu lệnh `switch`. Nếu không có câu lệnh `break`, chương trình sẽ thực hiện các công việc từ `case` tương ứng cho đến hết.

Ví dụ: Giả sử biến x lưu giá trị tháng (từ 1 đến 12). Kiểm tra xem tháng đó thuộc quý nào.

- Cách 1:

```
switch (x) {
        case 1: alert("Quy 1"); break;
        case 2: alert("Quy 1"); break;
        case 3: alert("Quy 1"); break;
        case 4: alert("Quy 2"); break;
        case 5: alert("Quy 2"); break;
        case 6: alert("Quy 2"); break;
        case 7: alert("Quy 3"); break;
        case 8: alert("Quy 3"); break;
        case 9: alert("Quy 3"); break;
        default: alert("Quy 4");
}
```

- Cách 2:

```
switch (x) {
        case 1:
        case 2:
        case 3: alert("Quy 1"); break;
        case 4:
        case 5:
        case 6: alert("Quy 2"); break;
        case 7:
        case 8:
        case 9: alert("Quy 3"); break;
        default: alert("Quy 4");
}
```

4. Toán tử điều kiện

Toán tử điều kiện[6] (*conditional op.*) là một toán tử cho ra 1 giá trị tùy thuộc vào biểu thức điều kiện là đúng hay sai.

Cú pháp:

```
<biểu thức điều kiện> ? <giá trị A> : <giá trị B>;
```

Trong cú pháp trên, nếu biểu thức điều kiện đúng thì kết quả là giá trị A, ngược lại là giá trị B. Giá trị A và B có thể là hằng số hoặc biểu thức.

Ví dụ: Tính giá trị của hàm số f(x), biết rằng:

$$f(x) = \begin{cases} 7x + 2 \text{ nếu } x \geq 0 \\ x^2 - 6 \text{ nếu } x < 0 \end{cases}$$

```
int fx = (x >= 0) ? (7 * x + 2) : (x * x - 6);
```

Lưu ý: Toán tử điều kiện có thể được viết bằng câu lệnh `if-else`. Ngược lại, một số câu lệnh `if-else` cũng có thể được viết bằng toán tử điều kiện.

V. Tương tác với trang web

1. Truy cập đến các đối tượng HTML

Trong JS, đối tượng[7] `document` tượng trưng cho toàn bộ trang web.

Để truy cập được những đối tượng trong trang web, JS cung cấp các phương thức[8] sau dành cho đối tượng `document`:

- `getElementById()`: Tìm thẻ HTML theo `id`:
  ```
  document.getElementById("demo");
  ```
- `getElementByClassName()`: Tìm thẻ HTML theo `class`:
  ```
  document.getElementByClassName("imgSkin");
  ```
- `getElementByTagName()`: Tìm thẻ HTML theo tên thẻ:
  ```
  document.getElementByTagName("p");
  ```
- `querySelector()` và `querySelectorAll()`: Tìm thẻ HTML theo selector CSS:
  ```
  document.querySelectorAll("p.inDam");
  ```

Ngoại trừ phương thức đầu tiên trả về 1 thẻ duy nhất, 3 phương thức còn lại sẽ trả về danh sách thẻ dưới dạng mảng.

2. Tương tác với các đối tượng HTML

Giả sử có thẻ HTML sau:

```
<a id="demo" href="http://www.google.com" style="color: red;">Hello</a>
```

Câu lệnh JS để truy cập được thẻ `<a>` này:

```
var e = document.getElementById("demo");
```

JS cho phép lấy và thay đổi nội dung, thuộc tính, định dạng CSS của thẻ theo những cách sau:

- Dùng thuộc tính `innerHTML` để lấy và thay đổi nội dung thẻ (nếu là control của form như textbox, checkbox… thì dùng thuộc tính `value`):
  ```
  document.write(e.innerHTML);
  e.innerHTML = "Xin chào";
  ```
- Dùng tên thuộc tính HTML để lấy và thay đổi giá trị thuộc tính HTML:
  ```
  document.write(e.href);
  ```

---

[6] Còn gọi là toán tử 3 ngôi

[7] Object. Hiện tại, hiểu đơn giản thì đối tượng nghĩa là biến (variable).

[8] Method. Hiện tại, hiểu đơn giản thì phương thức nghĩa là hàm (function).

```
e.href = "http://www.facebook.com";
```

- Dùng phương thức `setAttribute()` để thay đổi giá trị thuộc tính HTML:

```
e.setAttribute("href", "http://www.facebook.com");
```

- Dùng thuộc tính `style` để lấy và thay đổi giá trị thuộc tính CSS:

```
document.write(e.style.color);
e.style.color = "blue";
```

3. Xuất dữ liệu lên trang web

JS hỗ trợ 4 cách xuất dữ liệu lên trang web:

- Hiển thị trực tiếp vào thẻ HTML bằng thuộc tính `innerHTML` (xem phần III.2 ở trên).
- Hiển thị vào nội dung trang web bằng phương thức `document.write()`.
- Hiển thị bằng 1 cửa sổ popup bằng các phương thức `alert()`, `confirm()` và `prompt()`.
- Ghi vào console[9] của trình duyệt bằng phương thức `console.log()`.

VI. Xử lí sự kiện

Sự kiện (*event*) là 1 hành động được diễn ra khi người dùng tương tác với 1 đối tượng nào đó của trang web. Các đối tượng HTML hầu như đều hỗ trợ các sự kiện để giúp lập trình viên cài đặt các đoạn mã JS phản hồi lại với các tương tác của người dùng.

Danh sách các sự kiện thường gặp trong trang web:

Tên event	Ý nghĩa
`blur`	Đối tượng bị mất focus
`change`	Nội dung của đối tượng bị thay đổi (thường áp dụng cho các control trong form)
`click`	Click vào 1 đối tượng
`dblclick`	Double-click vào 1 đối tượng
`focus`	Đối tượng có focus
`keydown`	Nhấn 1 phím
`keyup`	Thả 1 phím
`keypress`	Nhấn và thả 1 phím
`mousedown`	Nhấn chuột
`mouseup`	Thả chuột
`mouseover`	Con trỏ chuột đi vào 1 đối tượng
`mouseout`	Con trỏ chuột đi ra khỏi đối tượng
`mousemove`	Con trỏ chuột đi chuyển bên trong đối tượng
`submit`	Form được submit

Việc cài đặt các đoạn mã JS cho các sự kiện được gọi là xử lí sự kiện (*event handling*). Mỗi sự kiện đều có thể được cài đặt thông qua 1 thuộc tính tương ứng của đối tượng HTML. Tên thuộc tính này có dạng `on<tên event>`.

Ví dụ: Khi click vào 1 nút trên trang web, hiển thị dòng chữ "Xin chào!" bằng popup:

[9] Xem ở chế độ Developer (phím tắt F12)

- Cách 1:

```
<button onclick="alert('Xin chào');">Click me</button>
```

- Cách 2:

```
<script>
	function ShowMsg() {
		alert("Xin chào!");
	}
</script>
<button onclick="ShowMsg()">Click me</button>
```

VII. Bài tập

Thực hiện các bài tập sau đây, mỗi bài tập nằm trong 1 trang web riêng. Cho phép người dùng nhập dữ liệu vào trang web thông qua textbox và thực hiện tính toán thông qua button.

1. Nhập vào 3 số nguyên a, b, c. Tìm số lớn nhất trong 3 số.
2. Nhập vào 3 số thực. Hãy thay tất cả các số âm bằng trị tuyệt đối của nó.
3. Giải phương trình bậc nhất dạng ax + b = 0 với a và b nhập từ bàn phím.
4. Giải phương trình bậc 2 dạng $ax^2 + bx + c = 0$ với a, b, c nhập từ bàn phím và $a \neq 0$.
5. Nhập vào tháng, năm. Cho biết tháng đó có bao nhiêu ngày. Lưu ý: Năm có thể là năm thường hoặc năm nhuận (Năm nhuận là năm chia hết cho 4 nhưng không chia hết cho 100; hoặc là năm chia hết cho 400).
6. Thiết kế 1 trang web gồm 1 textbox để nhập họ tên và 1 button Đăng nhập. Sau khi nhập họ tên và nhấn nút Đăng nhập, hiển thị lời chào "Xin chào abc!" (thay abc bằng họ tên người dùng vừa nhập).
7. Thiết kế 1 trang web gồm 2 textbox để nhập tên tài khoản, mật khẩu và 1 button Đăng nhập. Yêu cầu:
   - Khi di chuyển chuột lên textbox Mật khẩu thì hiển thị mật khẩu ở dạng plain text, khi di chuyển chuột ra khỏi textbox Mật khẩu thì hiển thị mật khẩu ở dạng password như bình thường.
   - Sau khi nhập thông tin và nhấn nút Đăng nhập, kiểm tra nếu tên tài khoản là admin và mật khẩu là 123456 thì thông báo "Đăng nhập thành công", ngược lại thông báo "Đăng nhập thất bại".