



# Javascript Basic

**Trần Huy Hoàng**

Email: [huyhoang.tran6669@gmail.com](mailto:huyhoang.tran6669@gmail.com)

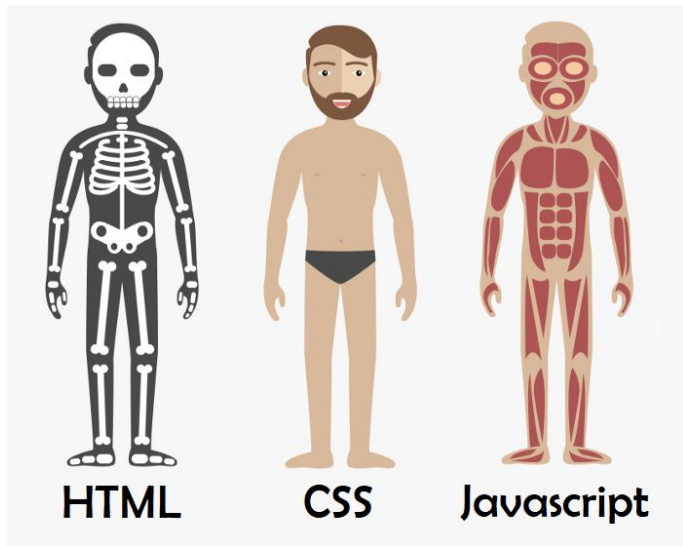
Phone: 0788.719.666

# Nội dung



1. Giới thiệu chung
2. Chương trình đầu tiên
3. Tìm hiểu biến trong Javascript
4. Kiểu dữ liệu trong Javascript
5. Hàm trong Javascript
6. Câu lệnh điều khiển trong Javascript
7. Vòng lặp trong Javascript

# 1. Giới thiệu chung



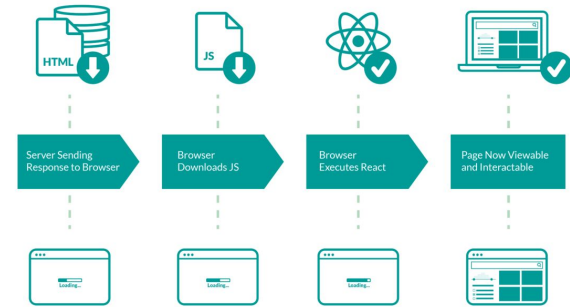
- ❖ **Javascript (JS)** là một ngôn ngữ kịch bản phía Client.
  - ❖ **Javascript** được sử dụng rộng rãi trong việc kết hợp với HTML/CSS để thiết kế web động.
- 
- **HTML** để xác định nội dung của các trang website.
  - **CSS** là ngôn ngữ tạo phong cách cho trang website.
  - **JavaScript** để lập trình hành vi của các trang website (tạo trang web tương tác).

# 1. Giới thiệu chung

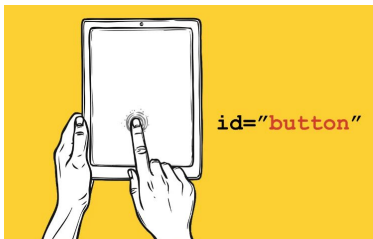


## 2. Phát triển mô hình web client rendering

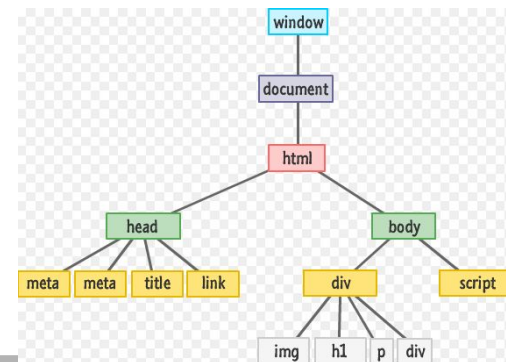
## CSR



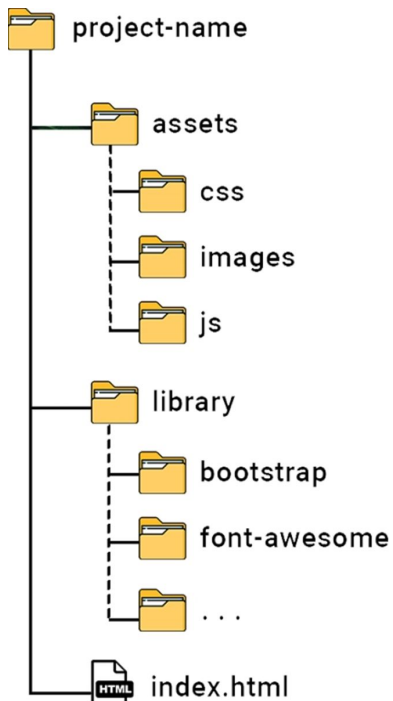
## 3. Kiểm soát các sự kiện, xử lý sự kiện tại biên



## 1. Tác động vào các thành phần của trang web



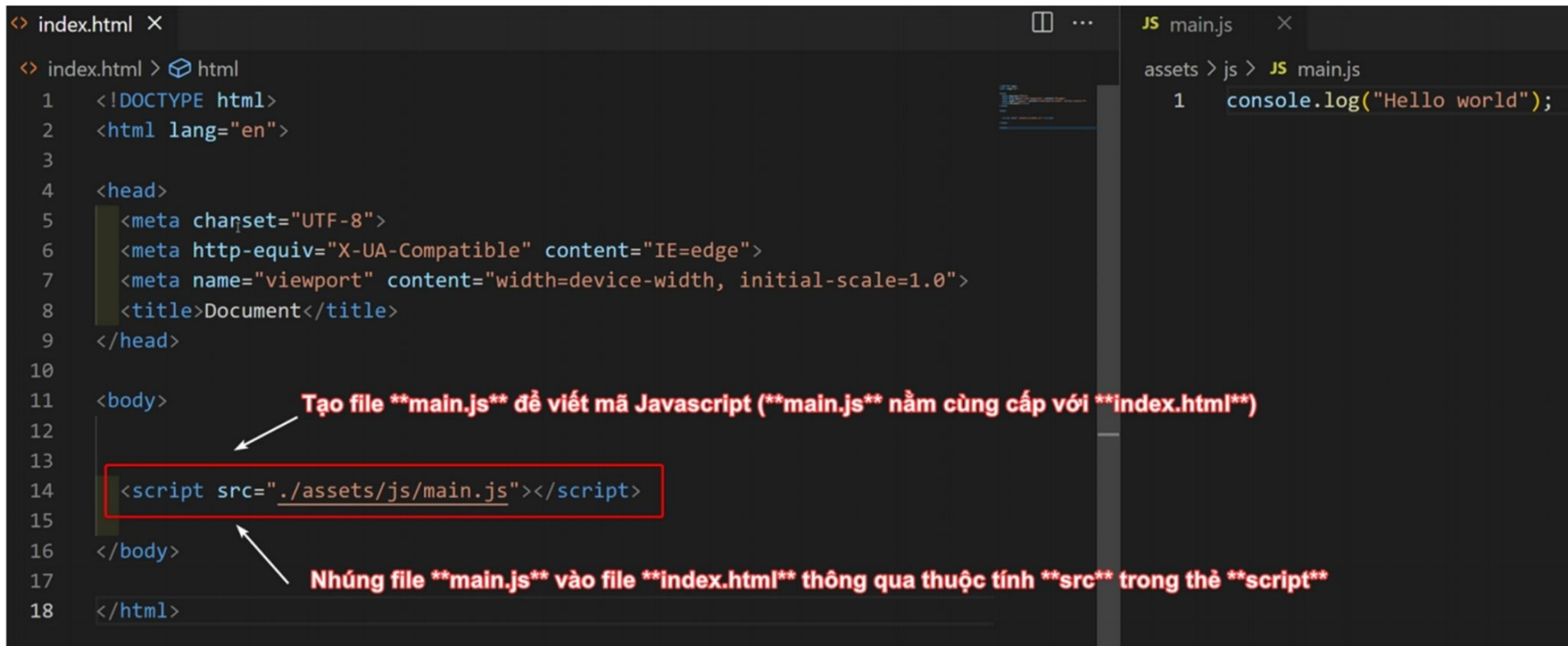
## 2. Chương trình đầu tiên



Tổ chức folder source code

- **project-name:** Tên thư mục dự án.
  - **assets:** Chứa folder css, images, js.
    - **css:** Chứa các file .css.
    - **images:** Chứa các file ảnh.
    - **js:** Chứa các file javascript.
  - **library:** Lưu trữ các file từ thư viện thứ 3.
  - **index.html**

## 2. Chương trình đầu tiên



```
<> index.html X
<> index.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Document</title>
9 </head>
10
11 <body>
12
13
14   <script src="./assets/js/main.js"></script>
15
16 </body>
17
18 </html>
```

```
JS main.js X
assets > js > JS main.js
1 console.log("Hello world");
```

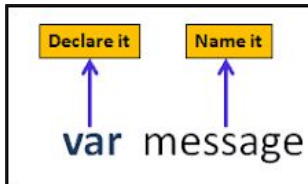
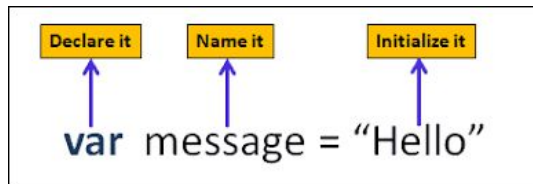
Tạo file **main.js** để viết mã Javascript (**main.js** nằm cùng cấp với **index.html**)

Nhúng file **main.js** vào file **index.html** thông qua thuộc tính **src** trong thẻ **script**

# 3.1 Tìm hiểu biến trong Javascript



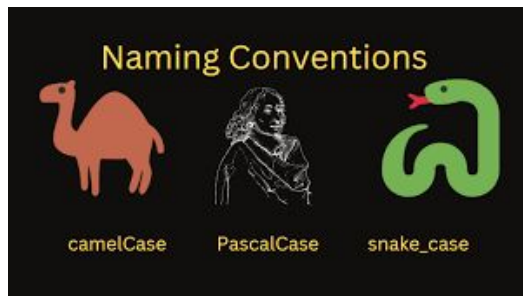
- **Biến** là một định danh dùng để lưu trữ dữ liệu, thông qua biến chúng ta có thể thực hiện các thao tác với dữ liệu.
- Mỗi biến có một kiểu dữ liệu riêng, dựa vào kiểu dữ liệu của biến có các thao tác khác nhau với biến.



Để khai báo biến sử dụng từ khóa : **let / const**

Sử dụng 2 cách sau để khai báo biến:

- Khai báo biến và không gán giá trị: **let age;**
- Khai báo biến và gán giá trị: **let age = 35;**



## Quy tắc đặt tên biến

- Chứa các ký tự chữ, số, `_`, `$`, ký tự đầu tiên không được là số
- Có phân biệt hoa thường
- Không trùng với các từ khóa của Javascript như: `for`, `while`, `this`, ..
- Nên đặt tên theo kiểu `camelCase` nếu tên biến có độ dài 2 từ trở lên cho dễ đọc.

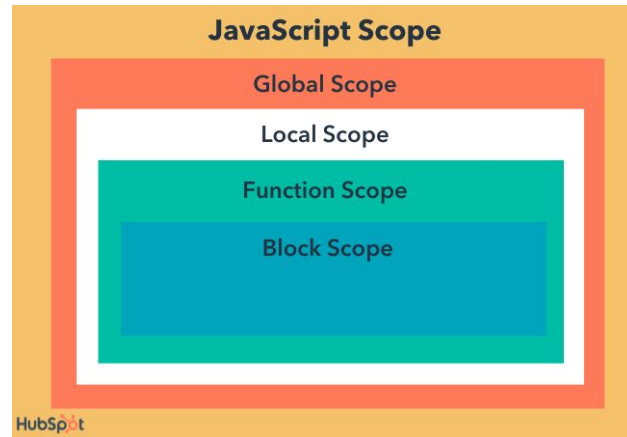
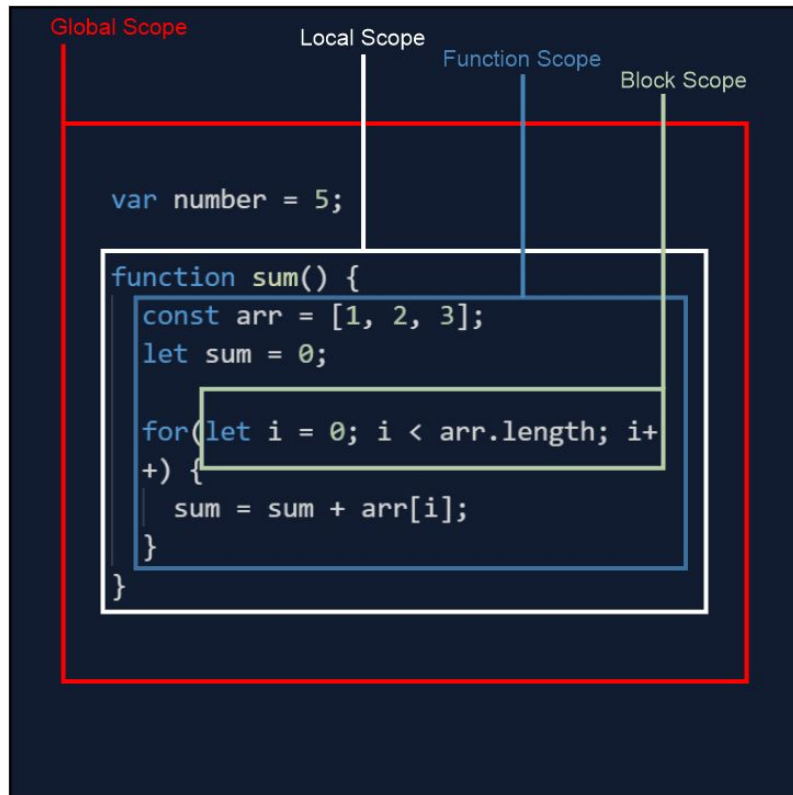
## LAB: Tìm các biến hợp lệ trong danh sách sau

1. \$usd
2. test123
3. username
4. docker7890-
5. \_self
6. 1a
7. string
8. first\_name
9. let
10. variable
11. Object
12. my-name
13. can't
14. error
15. Person
16. numBer

1. \$usd
2. test123
3. username
4. docker7890-
5. \_self
6. 1a
7. string
8. first\_name
9. let
10. variable
11. Object
12. my-name
13. can't
14. error
15. Person
16. numBer



## 3.2 Phạm vi của biến trong



- **Global scope:** Biến đó được định nghĩa bên ngoài function
- **Function scope:** Biến đó được định nghĩa bên trong function
- **Block scope:** Biến đó được định nghĩa bên trong cặp dấu {}

## 3.3 Phân biệt var, let, const trong javascript

**LAB:** Chạy đoạn code sau có lỗi không?

```
console.log(name);  
var name = 'Ky';
```



**< Hoisting? />**

**JS**

## 3.3 Phán biệt var, let, const trong javascript

	SCOPE	Hoisting	Tekrar Tanımlama	Değerini Değiştirme
Var	Function	✓ <small>Her yerde kullanılabilir</small>	✓	✓
Let	Block	! <small>İlk defa tanımlanır</small>	✗	✓
Const	Block	! <small>İlk defa tanımlanır</small>	✗	✗

Var

var apple = 🍏

!kutudaki "apple" adındaki bir şey

apple = 🍏

!öğeyi daha sonra değiştirebilirsiniz

Let

let apple = 🍏

!kacama kavanoz kutusunu içindeki "apple"

apple = 🍏

!öğeyi yalnızca kavanoz içinde değiştirebilirsiniz

Const

const apple = 🍏

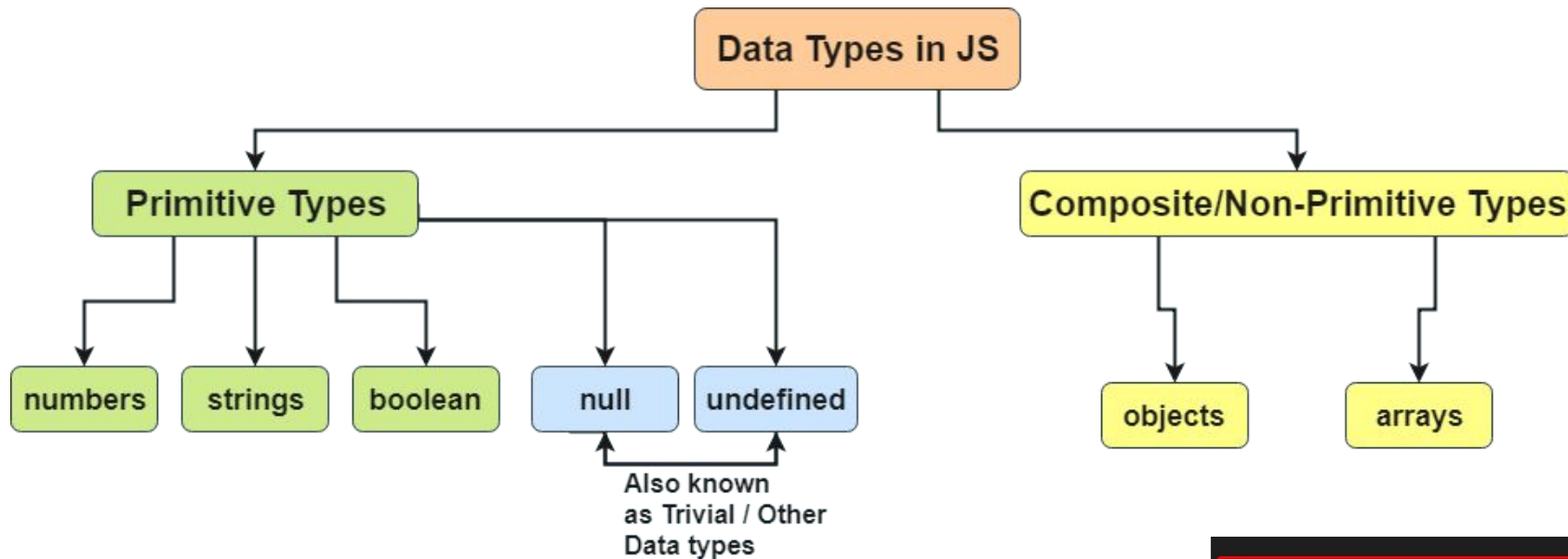
!kilitli kafeste kilitli bir "apple"

apple = 🍏

apple.multiply(3)

!öğeyi daha sonra değiştiremezsiniz

## 4. Kiểu dữ liệu trong Javascript



Để kiểm tra kiểu dữ liệu của 1 biến, chúng ta sử dụng toán tử **typeof**

```
let name = "Thuan nguyen";  
console.log(typeof name);
```

String

## 4.1. String

**String** (chuỗi) là một đoạn text có thể có một hoặc nhiều ký tự.

Các chuỗi đều phải được bao quanh bằng cặp dấu nháy đơn ' hoặc nháy kép "

```
let name = "Nguyễn Văn A";  
let email = "abc@gmail.com";
```

Trường hợp trong chuỗi cũng có xuất hiện dấu nháy đơn hoặc nháy kép thì phải thêm ký tự \ đằng trước dấu nháy đó.

```
let message = "Xin chào các bạn, mình tên là \"Bùi Hiên\". Mình sinh năm 1997";
```

Khi bạn muốn Enter xuống hàng một chuỗi thì bắt buộc phải sử dụng dấu + để nối chuỗi.

```
let message = "Xin chào các bạn, " +  
"mình tên là \"Bùi Hiên\". " +  
"Mình sinh năm 1997";
```

### Nối chuỗi trong Javascript

Để nối chuỗi chúng ta sử dụng dấu + để ghép hai chuỗi (hoặc biến) lại với nhau.

## 4.1 Number


**Number** (số) là một tập hợp của các con số (0 – 9) không chứa dấu khoảng trắng và có thể chứa dấu trừ (-) nằm ở đầu để đại diện cho số âm.

```
let age = 25;  
let num = 5.1;
```

## 4.2 Template string ES6

**Khai báo chuỗi sử dụng ký tự back-tick `` thay cho ký tự nháy đơn hoặc nháy kép.**

```
let name = "Bích Phương"  
let age = 2003
```

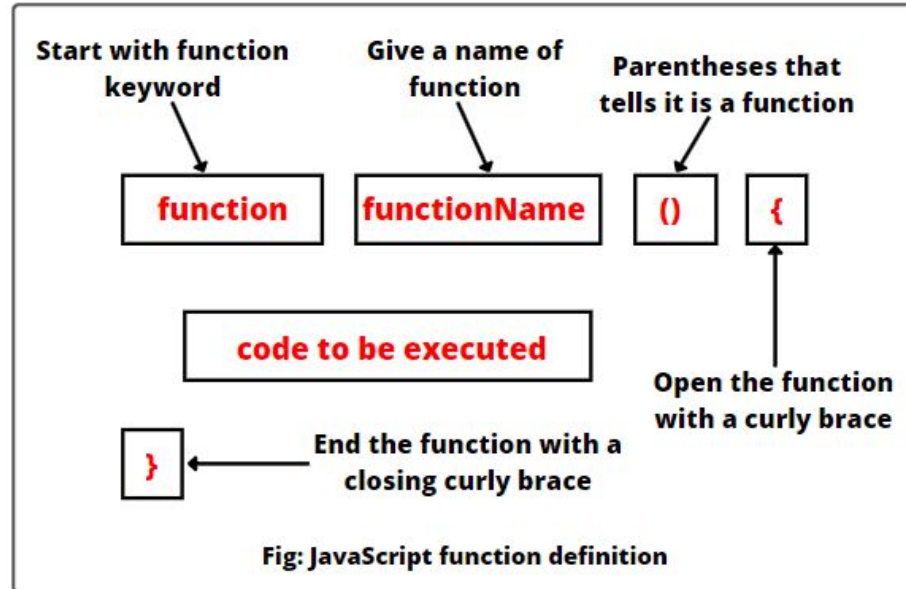
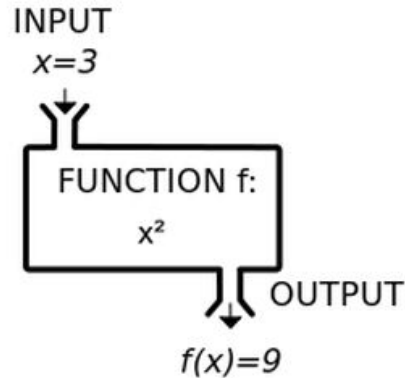


```
console.log(`Xin chào các bạn, mình tên là ${name}. Năm nay ${2022 - year} tuổi`);
```

**Khi khai báo chuỗi kiểu này có thể viết chuỗi trong nhiều dòng và có thể dùng dấu nháy đơn hoặc nháy kép thoải mái mà không cần dùng escape character.**

## 5. Hàm trong Javascript

**Function** (hàm) là tập hợp các đoạn code dùng để thực hiện một tác vụ cụ thể nào đó





## 5.1 Tham số && đối số

Phân biệt tham số (**parameter**) và đối số (**argument**)

- ❖ Tham số (parameter) là biến trong khai báo hàm.
- ❖ Đối số (argument) là giá trị thực của biến này được truyền vào hàm

The diagram shows a function definition and a function call on a dark background. The function definition is `function sum(param1, param2){` followed by `return param1 + param2;` and a closing brace. An orange bracket labeled "Parameters" points to `param1` and `param2` in the function signature. Below it, the function call is `sum(5, 6);`. A blue bracket labeled "Arguments" points to the values `5` and `6` passed to the function.

```
function sum(param1, param2){  
    return param1 + param2;  
}
```

Parameters

```
sum(5, 6);
```

Arguments

## 5.2 Các loại function thường dùng

- ❖ Function không có tham số
- ❖ Function có tham số
- ❖ Function có trả về kết quả
- ❖ Function không trả về kết quả

Đối với function không có từ khóa **return** thì coi như function đó không trả về kết quả

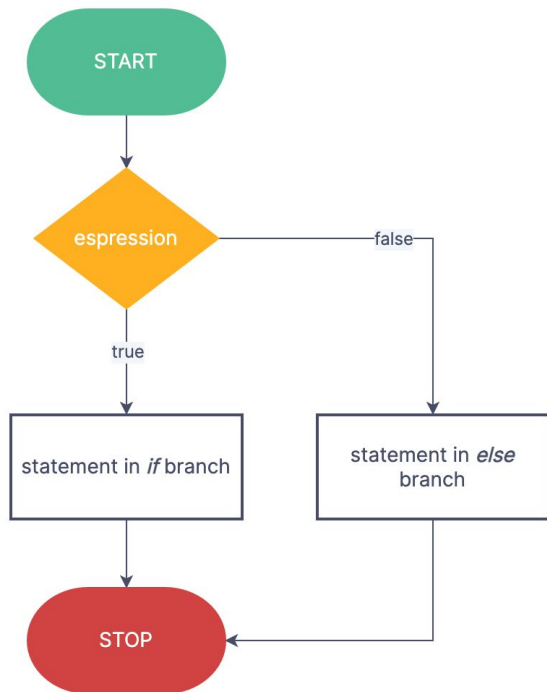
```
function sayHello(name) {  
    console.log(`Xin chào ${name}`);  
}  
sayHello("Bùi Hiền");
```

```
function sayHello() {  
    console.log("Xin chào các bạn");  
}  
  
sayHello();
```

```
function sum(a, b) {  
    return a + b;  
}
```

# 6.1 Câu lệnh điều khiển IF- ELSE trong Javascript

## Logical Operators



Operator	Meaning	Example	Result
&&	Logical and	$(5 < 2) \&\& (5 > 3)$	False
	Logical or	$(5 < 2)    (5 > 3)$	True
!	Logical not	$!(5 < 2)$	True

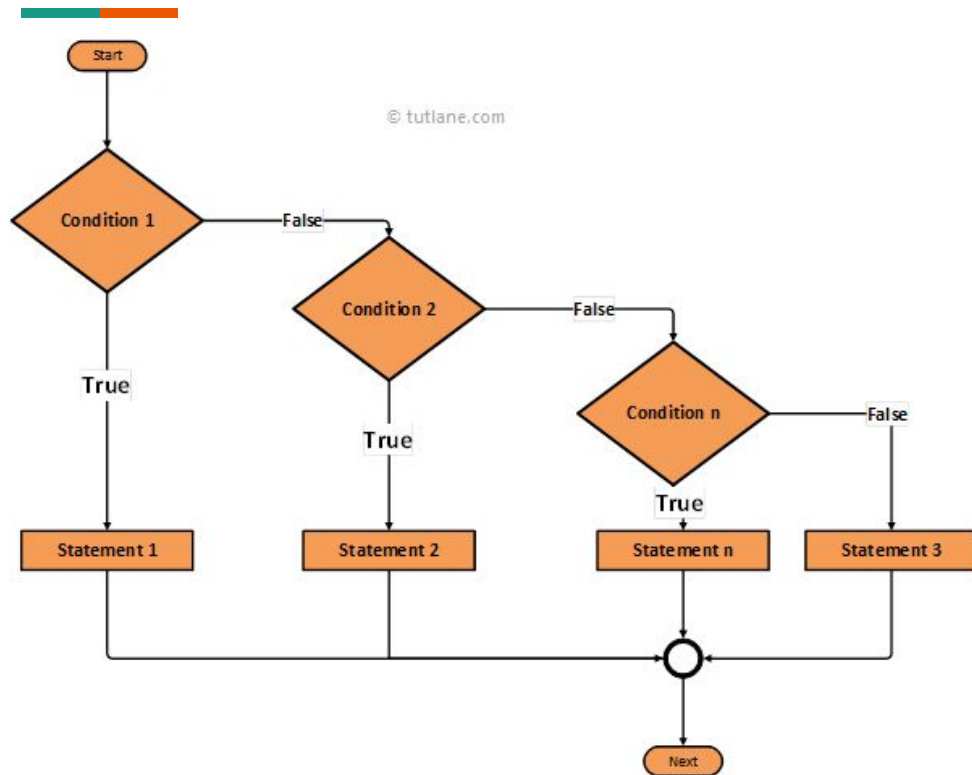
**Lab1:** Viết function truyền vào 2 số a, b. In ra màn hình số có giá trị lớn hơn.

**Lab2:** Viết function nhập vào 1 số. Kiểm tra số đó là số chẵn hay số lẻ.

**Lab3:** Viết function nhập vào 1 số. Kiểm tra số đó có đồng thời chia hết cho 3 và 5 không.

**Lab4:** Viết function nhập vào 3 số a, b, c. Kiểm tra xem c có bằng  $a + b$  không?

## 6.2 Câu lệnh điều khiển IF-ELSE lồng nhau

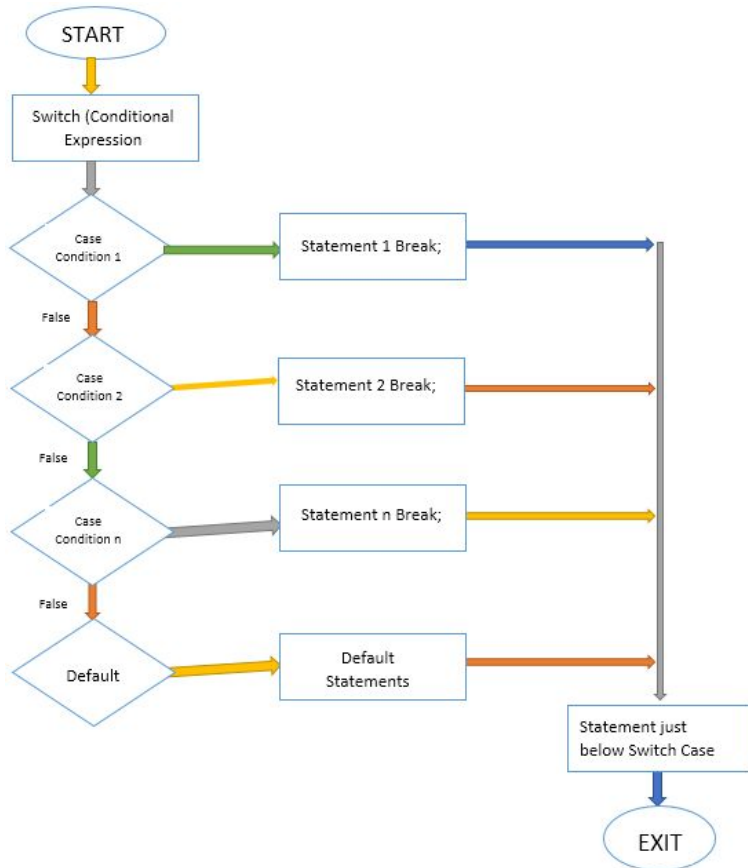


**Lab5:** Viết function nhập vào biến mark có giá trị từ 0 -> 100. Kiểm tra giá trị của biến mark và in ra nội dung sau

- "A" nếu  $\text{mark} \geq 85$
- "B" nếu  $70 \leq \text{mark} < 85$
- "C" nếu  $40 \leq \text{mark} < 70$
- Các trường hợp còn lại in ra "D"

**Lab6:** Viết function nhập vào 1 số. Kiểm tra số đó có đồng thời chia hết cho 3 và 5 không.

## 6.3 Câu lệnh điều khiển Switch - case



```
// program using switch statement
let a = 2;

switch (a) {

    case 1:
        a = 'one';
        break;
    case 2:
        a = 'two';
        break;
    default:
        a = 'not found';
        break;
}

console.log(`The value is ${a}`);
```

## 6.3 Câu lệnh điều khiển Switch - case

Giả sử khi đến một quán nước, ở đó có một cái menu giống như bên dưới và trong tay bạn chỉ có đúng **\*\*10.000\*\* VND**

**Câu hỏi:** Nếu yêu cầu bạn chọn một món nước uống có giá bằng đúng với số tiền mà bạn đang có, thì món nước uống đó là món gì ?

MENU		
Cà phê sữa	.....	12.000đ
Cà phê đá	.....	10.000đ
Sting dâu	.....	8.000đ
Trà đá	.....	2.000đ

```
let money = 10000

switch (money) {
  case 12000: {
    console.log("Cà phê sữa");
    break;
  }
  case 10000: {
    console.log("Cà phê đá");
    break;
  }
  case 8000: {
    console.log("String dâu");
    break;
  }
  case 2000: {
    console.log("Trà đá");
    break;
  }
  default: {
    console.log("Không có đồ uống phù hợp");
    break;
  }
}
```

## 6.3 Câu lệnh điều khiển Switch - case



**Lab1:** Tạo biến day có giá trị từ 0 -> 6, là các ngày trong tuần Trong đó: (0 - chủ nhật, 1 - thứ 2, ..., 6 - Thứ 7) Sử dụng switch-case để in ra ngày tương ứng với giá trị của biến day

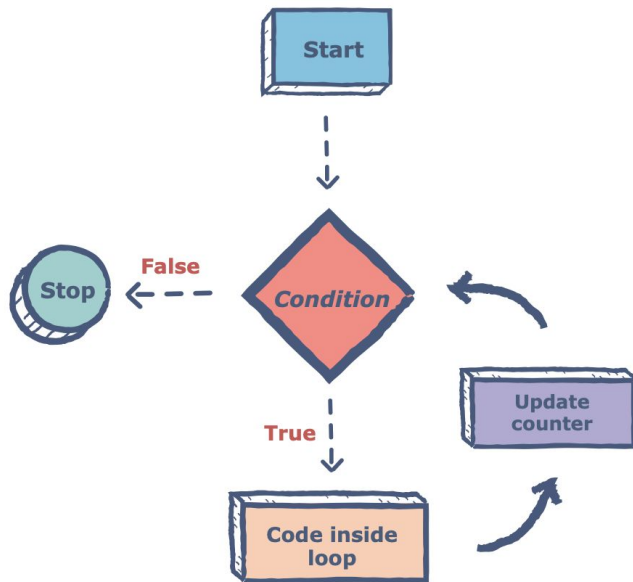
Ví dụ: day = 6 --> “Hôm nay là thứ 7”

**Lab2:** Tạo biến month có giá trị từ 1 -> 12, là các tháng trong năm

Sử dụng switch-case để in ra mùa tương ứng với giá trị của biến month

- 1, 2, 3 : Mùa xuân
- 4, 5, 6 : Mùa hạ
- 7, 8, 9 : Mùa thu
- 10, 11, 12 : Mùa đông

## 7. Vòng lặp trong Javascript



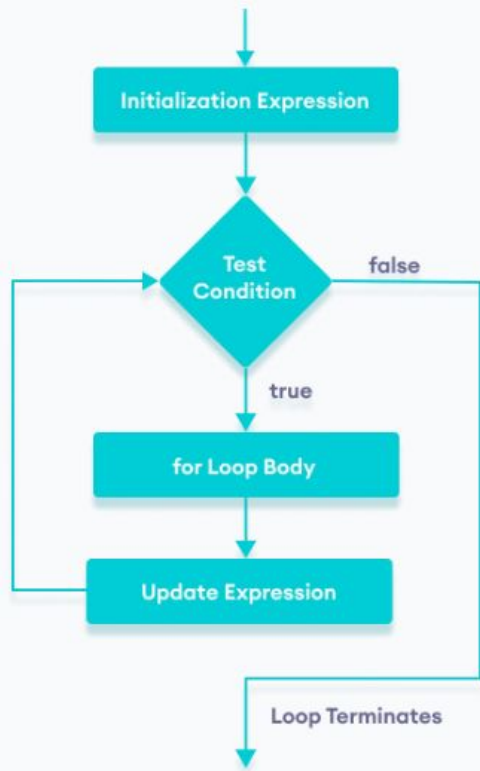
**Vòng lặp** được sử dụng để thực thi một đoạn code lặp đi lặp lại một số lần nhất định

Một số loại vòng lặp trong Javascript:

- for
- while
- do/while
- forEach



## 7.1 For Loop

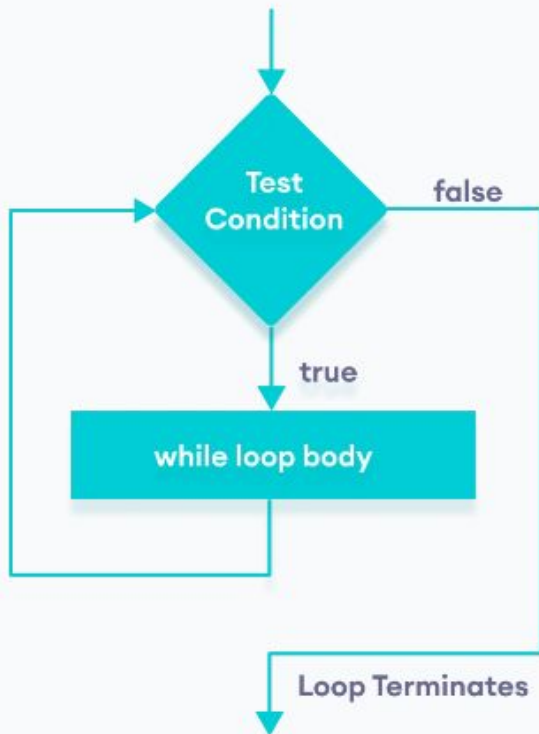


```
for (initialExpression; condition; updateExpression) {  
  // for loop body  
}
```

```
// program to display text 5 times  
const n = 5;  
  
// looping from i = 1 to 5  
for (let i = 1; i <= n; i++) {  
  console.log(`I love JavaScript.`);  
}
```

**Lab1:** Viết function truyền vào 1 chuỗi bất kỳ, hãy viết hàm có tác dụng sao chép đó chuỗi lên 10 lần, ngăn cách nhau bởi dấu gạch ngang

## 7.2 While Loop

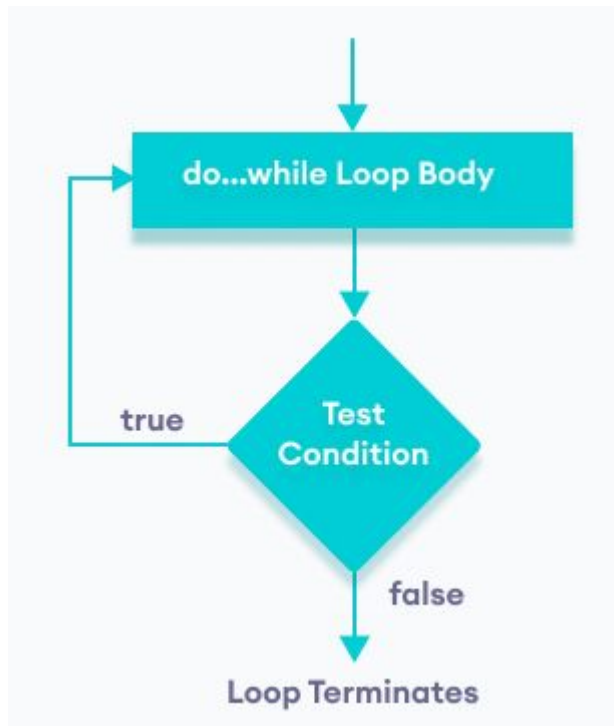


```
while (condition) {  
    // body of loop  
}
```

```
// program to display numbers from 1 to 5  
// initialize the variable  
let i = 1, n = 5;  
  
// while loop from i = 1 to 5  
while (i <= n) {  
    console.log(i);  
    i += 1;  
}
```

**Lab1:** Viết function truyền vào 1 chuỗi bất kỳ, hãy viết hàm có tác dụng sao chép đó chuỗi lên 10 lần, ngăn cách nhau bởi dấu gạch ngang

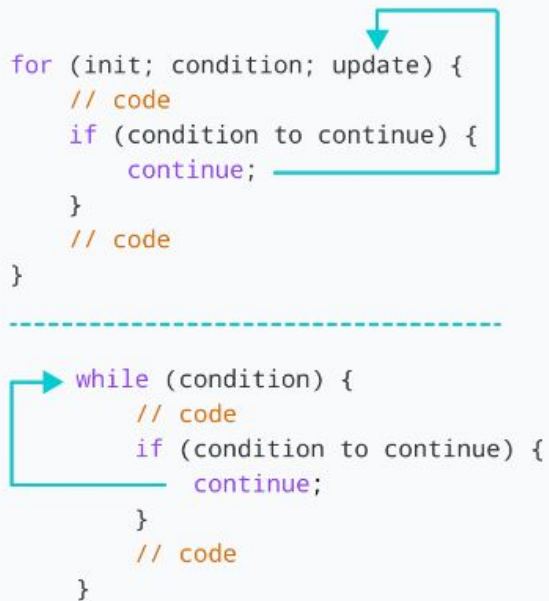
## 7.3 Do While Loop



```
do {  
    // body of loop  
} while(condition)
```

```
// program to display numbers  
let i = 1;  
const n = 5;  
  
// do...while loop from 1 to 5  
do {  
    console.log(i);  
    i++;  
} while(i <= n);
```

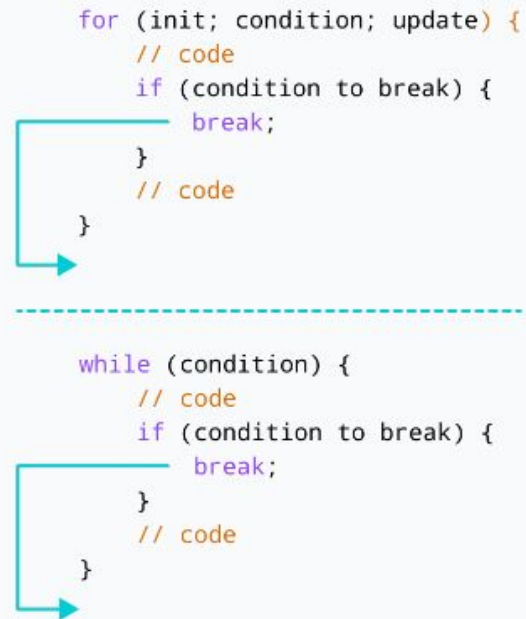
## 7.4 Continue & break trong vòng lặp



```
for (init; condition; update) {  
    // code  
    if (condition to continue) {  
        continue;  
    }  
    // code  
}
```

---

```
while (condition) {  
    // code  
    if (condition to continue) {  
        continue;  
    }  
    // code  
}
```



```
for (init; condition; update) {  
    // code  
    if (condition to break) {  
        break;  
    }  
    // code  
}
```

---

```
while (condition) {  
    // code  
    if (condition to break) {  
        break;  
    }  
    // code  
}
```

## 7.4 Continue & break trong vòng lặp

**Lab1:** Giá trị được in ra màn hình của hàm số sau là bao nhiêu?

```
// program to print the value of i
for (let i = 1; i <= 5; i++) {

    // condition to continue
    if (i == 3) {
        continue;
    }

    console.log(i);
}
```

```
// program to print the value of i
for (let i = 1; i <= 5; i++) {
    // break condition
    if (i == 3) {
        break;
    }
    console.log(i);
}
```

# Lab



**Lab1:** Viết function truyền vào 1 chuỗi bất kỳ và 1 số nguyên dương  $n > 1$ , hãy viết hàm có tác dụng sao chép đó chuỗi lên  $n$  lần, ngăn cách nhau bởi dấu gạch ngang.

**Lab2:** Tính tổng các số chia hết cho 5 từ 0 -> 100

**Lab3:** Viết hàm truyền vào 2 số nguyên, tính tổng tất cả các số nguyên nằm giữa chúng.

**Lab4:** Cho 1 số, kiểm tra xem số đó có phải là số nguyên tố hay không, kết quả trả về true hoặc false.

**Lab5:** Cho 1 số nguyên dương bất kỳ. Tính tổng tất cả các số nguyên tố mà nhỏ hơn hoặc bằng tham số truyền vào.

**Lab6:** Cho 1 số nguyên dương, viết hàm tính tổng tất cả các ước số của số đó.

**The End**