Day 10: JS

Phần 1: Giới thiệu về JavaScript?
Phần 2: Biến trong Js
Phần 3: Kiểu dữ liệu trong Js

Phần 1: Giới thiệu về JavaScript?

1: Javascript là gì ?

- Javascript là ngôn ngữ lập trình bậc cao, cực kỳ linh hoạt được sử dụng chủ yếu để tạo ra ứng dụng chạy trên trình duyệt web.
- Được sử dụng bởi nhiều công ty lớn trên thế giới
- Có thể được sử dụng để lập trình FrontEnd , Backend
- Có thể được tạo ra nhiều ứng dụng khác nhau như Web, Mobile Apps, Games…

  +Web App: ReactJS, Vuejs , Angular,Nodejs,...

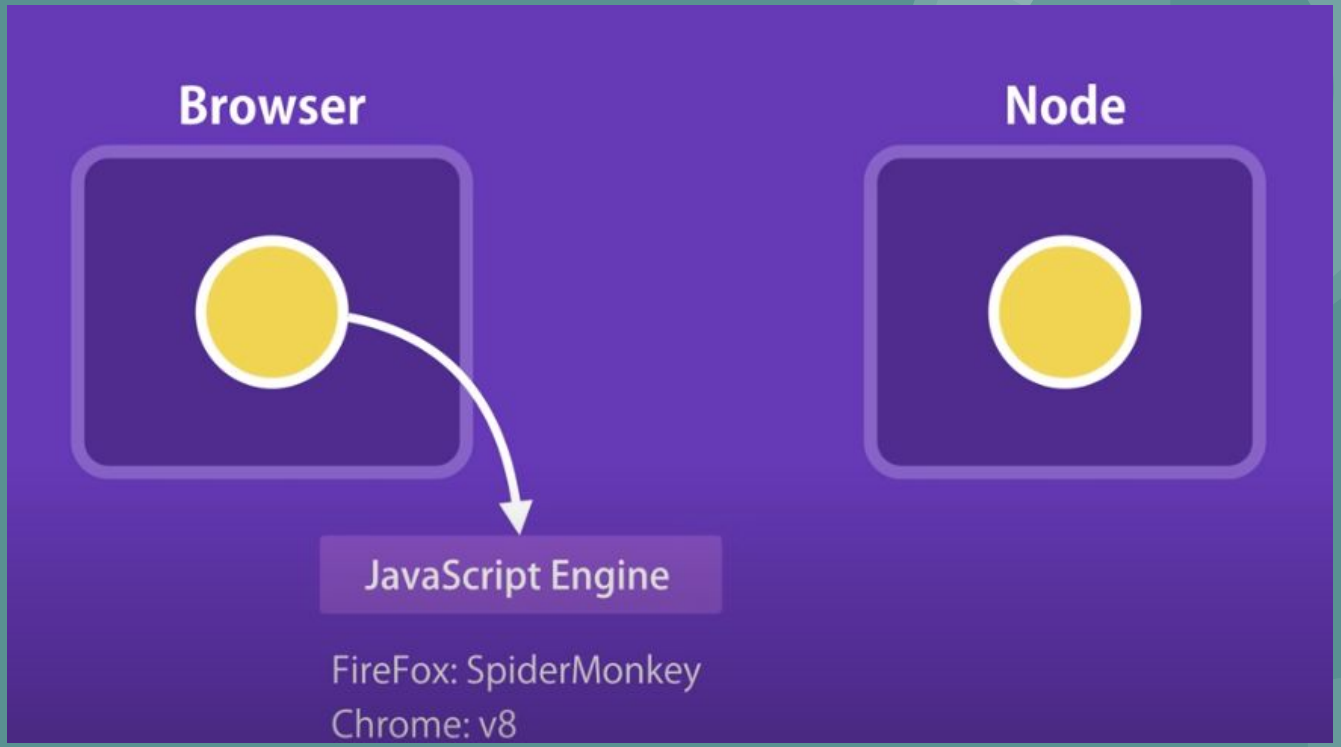
  +Mobile App: React Native,Ionic

  +Desktop App: Electron …

  +Graphic/Game: JS ThreeJS ,Phaser …

Phần 1: Giới thiệu về JavaScript?

2: Javascript Engine



Phần 1: Giới thiệu về JavaScript?

3:Tại sao nên học Javascript.

- Ngôn ngữ phổ lập trình phổ biến nhất.
- Dễ học hơn so với các ngôn ngữ khác
- Sử dụng để lập trình FrontEnd , Backend
- Tích hợp sẵn trong hầu hết các trình duyệt

4: Javascript viết ở đâu.

- Thẻ script trong html
- Các file js được dẫn tới bởi file html (giống với css)
- Nên để code js ở cuối cùng của file html (trong body)
    - Cải thiện trải nghiệm người dùng khi có nhiều code js cần load
    - Đảm bảo code js được chạy khi html đã được load xong
- Trong thực tế, nên chia các file js theo công việc mà chúng thực hiện

Phần 2: Biến trong Js

1:Khái niệm biến là gì?

-Giống như hầu hết các ngôn ngữ lập trình khác, biến là nơi ta lưu trữ tạm thời một dữ liệu gì trong bộ nhớ máy tính. **có thể** thay đổi được

-Giống như khi ta cất một món đồ nào đó vào trong những chiếc hộp và gán nhãn cho chúng để dễ dàng tìm kiếm khi cần

- Chiếc hộp là bộ nhớ máy tính
- Đồ trong hộp là thông tin ta lưu vào biến
- Nhãn là tên mà chúng ta đặt cho biến
- Ví dụ: const name = “nghiem”; let age = 25; let address;
- Mặc định biến có giá trị là undefined , giống như chiếc hộp chưa có đồ,

Phần 2: Biến trong Js

2:Cách khai báo biến

-có 3 cách khai báo biến,bao gồm các từ khóa var,let,const

+lưu ý: từ khóa let và const xuất hiện từ phiên bản ES6(2015), 2024 rồi không ai dùng var (đã quá lỗi thời rồi và không nên sử dụng)

Cú pháp khai báo biến như sau: let <tên biến>;

Ví dụ câu lệnh sau khai báo một biến có tên là "name":

```
let name;
name = “nghiêm”
```

=>Nghĩa là biến "name" đang liên kết đến vùng nhớ có giá trị là "Nghiêm".

Phần 2: Biến trong Js

3:Cách thay đổi giá trị của biến số

-Để thay đổi giá trị của biến, bạn chỉ cần gán giá trị mới cho nó, ví dụ:

Phần 2: Biến trong Js

-Hoặc bạn cũng có thể gán giá trị của biến số này cho biến số khác, ví dụ:

```
// Khai báo biến và gán giá trị ban đầu
let language1 = "JavaScript";
console.log(language1);    'JavaScript'
// Khai báo biến số 2
let language2;
// Gán giá trị của biến language1 cho biến language2
language2 = language1;
console.log(language2);    'JavaScript'
// Giá trị của biến language1 vẫn không đổi
console.log(language1);    'JavaScript'
```

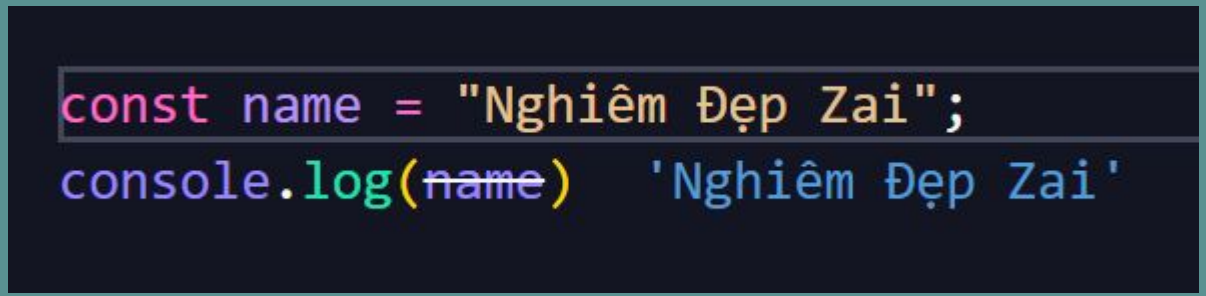
Phần 2: Biến trong Js

4: const (hằng) trong js

-Hằng là "tên biểu tượng" đại diện cho một giá trị không thay đổi trong chương trình.
Để khai báo hằng trong JavaScript, bạn dùng từ khóa const với cú pháp: const <tên hằng> = <giá trị của hằng>;.

Với chú ý là: bạn bắt buộc phải gán giá trị cho hằng NGAY khi khai báo.

-Cách khai báo hằng trong JavaScript
const <tên hằng> = <giá trị của hằng>;



Phần 2: Biến trong Js

5: Phạm vi truy hoạt động biến (Scope)

-Có 3 loại scope trong js

- **Global Scope (Phạm vi toàn cục):** Biến được khai báo trong global scope có thể được truy cập từ bất kỳ đâu trong mã nguồn JavaScript. Chúng tồn tại trong suốt quá trình chạy của chương trình.
- **Local hay Function Scope (Phạm bị hàm):**Biến khai báo trong một hàm chỉ tồn tại trong phạm vi của hàm đó.
- **Block scope (phạm vi khối):** Với giới thiệu của từ khóa "let" và "const" trong ES6, biến khai báo bằng "let" hoặc "const" được giới hạn trong phạm vi của khối mã mà chúng được khai báo. Một khối mã là một nhóm câu lệnh được bao quanh bởi cặp dấu ngoặc nhọn "{}", chẳng hạn như trong một vòng lặp "for" hoặc một câu lệnh "if".

=>Sử dụng phạm vi (scope) đúng cách là một yếu tố quan trọng trong việc viết mã JavaScript hợp lý, đảm bảo tính nhất quán và tránh xung đột giữa các biến. 😊

-**Biến toàn cục** là những biến được khai báo bên ngoài scope của đoạn code đang xét
-**Biến cục bộ** là những biến được khai báo bên trong scope của đoạn code đang xét

Phần 2: Biến trong Js

6:Hoisting

- Hoisting là gì?: khi bạn khai báo 1 biến hoặc 1 hàm, JS sẽ “kéo” các khai báo lên trên đầu phạm vi (Scope) của chúng trong quá trình thực thi.

-Tuy nhiên việc hoisting giữa var & let/const có sự khác nhau đôi chút:

  - Với var: khi hoisting sẽ được gán giá trị mặc định là undefined => có thể đọc được giá trị trước khi gán giá trị thực tế

  - Với let/const: Biến được đưa lên đầu phạm vi nhưng không được khởi tạo, dẫn đến lỗi nếu truy cập trước khi khai báo

```
console.log(a)  undefined
var a = 10;
```

```
console.log(a)  Cannot access 'a' before initialization
let a = 10;
```

Phần 2: Biến trong Js

7: Sự khác biệt giữa var - let -const

Đặc điểm	var	let	const
Phạm vi (scope)	*Hàm hoặc toàn cục*	*Block scope*	*Block scope*
Hosting	*Có (gán undefined)*	*Có (nhưng không khởi tạo)*	*Có (nhưng không khởi tạo)*
Tái khai báo	*Được phép*	*Không được phép*	*Không được phép*
Gán lại giá trị	*Được phép*	*Được phép*	*Không được phép (trừ object/array)*

Phần 2: Biến trong Js

8: Quy tắc đặt tiên biến.

- Không được trùng với những từ khoá của js
- Nên, rất nên đặt tên biến theo ý nghĩa của dữ liệu mà nó sẽ lưu (chứa),
- Ví dụ: không nên đặt tên hộp đồ chơi (toys) là rác (trash)
- Không được bắt đầu bằng số,
- Không được chứa khoảng trắng hoặc dấu “-”
- Các tên biến có phân biệt chữ hoa chữ thường
- Nên đặt theo camelCase

- Dùng const và let, không dùng var

III: Kiểu dữ liệu

Các kiểu dữ liệu.

-Kiểu dữ liệu tham trị

- String: “this is test string”
- Number: 9
- Boolean: true / false
- Null: không có giá trị , là một giá trị đặc biệt
- Undefined: chưa được gán giá trị, là một kiểu dl đồng thời là một giá trị đặc biệt

-Kiểu dữ liệu tham chiếu

- Array: lưu dữ liệu thuộc nhiều kiểu dưới dạng danh sách
- Object: lưu dữ liệu thuộc nhiều kiểu dưới dạng key-value
- Function

Phần 3: Kiểu dữ liệu

1:Kiểu dữ liệu string.

+String là kiểu dữ liệu dùng để biểu diễn chữ, văn bản, đoạn văn bản,...

+Có ba cách để biểu diễn string trong JavaScript:

+Dùng dấu nháy đơn (')

+Dùng dấu nháy kép (")

+Dùng dấu "backtick" (`)

+Dấu nháy đơn và dấu nháy kép là hoàn toàn giống nhau.

+Riêng với dấu "backtick", bạn có thể sử dụng biến, hằng hoặc thậm chí viết một biểu thức trong đó, với cú pháp ${...}

```
const string = "Hello";
const age = 30;
const isMarried = true;
const message = `${string} Nghiêm, Tôi ${age} Tuổi và ${
  isMarried ? "Đã kết hôn" : "Độc thân"
}`;

console.log(string);  'Hello'
console.log(message);  'Hello Nghiêm, Tôi 30 Tuổi và Đã kết hôn'
```

Phần 3: Kiểu dữ liệu

2:Kiểu dữ liệu number.

-Kiểu dữ liệu number là kiểu dữ liệu dạng số (tương tự trong toán học). Number trong JavaScript không có cú pháp gì đặc biệt. Bạn chỉ cần viết số ra.

-JavaScript có hai loại số là: số nguyên và số thực.

```
let n1 = 66; // số nguyên dương
let n2 = -66; // số nguyên âm
let n3 = 3.14; // số thực dương
let n4 = -3.14; // số thực âm
let n5 = 2e3; // => 2*10^3 = 2000
let n6 = 2e-3; // => 2*10^(-3) = 0.002
let n7 = 0xff; // số dạng hexa (hệ cơ số 16): 15*16 + 15 = 255
let n8 = 067; // số dạng octa (hệ cơ số 8): 6*8 + 7 = 55
let n9 = 0b11; // số dạng nhị phân (hệ cơ số 2): 1*2 + 1 = 3
```

Phần 3: Kiểu dữ liệu

+Ngoài những loại số trên, JavaScript còn có 3 số đặc biệt là: Infinity, -Infinity và NaN.

Infinity: là số dương vô cùng. Đây là giá trị đặc biệt và nó lớn hơn bất kỳ số nào khác

Infinity: là số âm vô cùng. Đây cũng là giá trị đặc biệt và nó nhỏ hơn bất kỳ số nào khác

NaN: là viết tắt của Not a Number, được sử dụng để đại diện cho những trường hợp tính toán sai hoặc kết quả của một phép tính không xác định

Phần 3: Kiểu dữ liệu

3: Kiểu dữ liệu boolean (kiểu logic).

+Boolean là kiểu dữ liệu logic chỉ bao gồm hai giá trị là true (đúng, chính xác) và false (sai, không chính xác),

- có 2 cách để nhận giá trị kiểu boolean.
- Gán giá trị trực tiếp
- Nhận kết quả từ 1 biểu thức khác

Phần 3: Kiểu dữ liệu

Kiểu dữ liệu boolean (kiểu logic).

- Khi thực hiện các toán tử so sánh và kiểm tra, một số dữ liệu sẽ được coi là truthy (tương đương true) và sẽ cho ra kết quả giống như true, một số sẽ được coi là falsy và sẽ cho ra kết quả như false
- falsy
  - false,
  - 0
  - ‘’
  - Null
  - Undefined
  - NaN
- Truthy
  - {}
  - []
  - ‘0’, 1,-1
  - ‘false’
  - function

Phần 3: Kiểu dữ liệu

4:Kiểu dữ liệu: null và undefined

- Null: là kiểu dữ liệu được gán cho biến, thường được hiểu là không biết (Không có)
- Undefined: hường thấy khi ta khai báo biến mà chưa gán giá trị cho nó, hoặc một hàm không trả về giá trị

Phần 3: Kiểu dữ liệu

5: Kiểu dữ liệu Object

- Giống như mọi đồ vật ở quanh ta, chúng có các thuộc tính để phân biệt với các đồ vật khác

- Sử dụng Object khi cần lưu trữ nhiều thông tin liên kết với nhau và có thể có nhiều kiểu dữ liệu
- Ví dụ: mỗi học sinh khi đăng kí học cần có các thông tin như Tên, tuổi
  - Sinh Viên 1: tên Nghiêm, Tuổi 20
  - Sinh Viên 2: tên Phong, Tuổi : 21
- Lưu trữ dữ liệu dưới dạng cặp key-value, cách nhau bởi dấu “,”

```
const student1 = {
    name: 'Nghiem',
    age: 20
}
const student2 = {
    name: 'Phong',
    age: 21
}
```

Phần 3: Kiểu dữ liệu

6: Kiểu dữ liệu Mảng [array]

- Mảng: được sử dụng để lưu trữ một danh sách các dữ liệu, có thể thuộc các kiểu dữ liệu khác nhau
- Cú Pháp: “[danh sách các dữ kiệu cách nhau bởi dấu ‘,’]”
- Ví dụ:
  - Danh sách các số chẵn
  - Danh sách học sinh trượt môn
- Truy cập đến từng phần tử thông qua chỉ số thứ tự của chúng trong mảng (bắt đầu từ 0)
- Ví dụ:
  - evenNumb[1] => 4
  - failStudent[0] => {id:1,name: ‘son’}

```
const evenNumb = [2, 4, 6, 8]
const failStudent = [{ id: 1, name: "Son" }, { id: 2, name: 'long' }]
```