**Kiến thức chung**

IDE: môi trường phát triển tích hợp, là 1 phần mềm giúp các ltv phát triển pm, gồm trình soạn thảo, biên dịch, trình gỡ lỗi, và 1 số công cụ tùy vào IDE

code

“use strict” : khắt khe về code, nếu chưa khai báo vẫn chạy đc nhưng về sau sẽ lôi, nếu thêm này vào sẽ báo lỗi. giống ngoại lệ. Hiện đại đã tự động dùng

biến

tên k trùng từ khóa, không bắt đầu bằng số, k dùng kí tự đặc biệt ngoại trừ $ \_

‘aa’ “aa” giống nhau

var - toàn cục, có thể khai báo lại

let - block, k thể khai báo lại cùng phạm vi ( khai báo lại khi cái đầu tiên ngoài block, nhưng thoát block thì giữ giá trị đầu tiên hoặc biến mất nếu chưa được khai báo toàn cục)

const - block , k thể khai báo lại. Dùng cho hằng số

KDL

number:

string: đặc biệt `${name}` : sẽ ra giá trị của biến name

bigint: số + n, 123n

boolean

null

undefined

symbol:

Toán tử

khá giống java

khác vs java gồm

\*\* vs ^: lũy thừa

khi tính toán string thì nó xem là number nếu là dạng number “2” - “1”

trừ dấu + thì nó ghép string, nếu là bigInt thì phải chung bigInt nếu k sẽ lỗi..

So sánh

khá giống java

khác so sánh bằng có thêm so sánh nghiêm ngặt

1==”1” true js tự hiểu “1” cũng là 1

1===”1” false khác KDL là false

tương tự != vs !==

this

arrow function, anonymous function(settimeout,…), event handle k có this, nó sẽ gọi “object global” -window.

Object 1. let o = { key1 : value, key2 : value}

2. function Person(name, age) { this.name = name;

this.age = age; this.greet = function() { console.log('Hello!'); }; }

*Khởi tạo từ khai báo bt, hàm khởi tạo, từ khóa new Object*

o.sayHello = function () { console.log("Hello!"); }; // tạo method cho object

Delete o.key1/o[“keyString”]

console.log("name" in user); // true/false

\_.isEqual(object1;object2); // so sánh giá trị

**Prototype** giúp kế thừa , giảm tiêu tốn bộ nhớ ( thay vì thiết kế function trong lúc định nghĩa thì mình tạo ở ngoài vẫn kế thừa được)

Method

Các loại hàm

default: function a() {return b;}

hàm khởi tạo object - ở trên mục KIẾN THỨC CHUNG ->Object

hàm biểu thức: let a = function (){return b;}

hàm biểu thức dùng arrow: **() => {} ,** *k có bối cảnh thực thi,k thể tạo construs, k có ‘this’, nó kế thừa this từ bên ngoài*

const a = (v,u) => {return b+2;}

const a = (v,u) => b + 2;

**Regex** – công cụ để xử lý, tìm kiếm, chỉnh sửa văn bản như kiểm tra có phải email hay sđt hợp lệ, kiểm ra là số hay là chữ,..

**DOM**

**Javascript worker** giúp lưu trên cookie, sử dung offline

**DOM**

**Bất đồng bộ (callback,promise,async/await)**

**Design pattern**

**ES6 - tiêu chuẩn hóa của js, cải tiến về mặt code**

1. let, const
2. template literal `${name}` name là biến, hoặc biểu thức
3. Arrow function (dữ ngữ cảnh của ‘this’ trong phạm vi chứa nó)

// Function thông thường

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

// Arrow function

const addArrow = (a, b) => a + b;

1. có thể truyền giá trị mặc định cho tham số
2. *classes*
3. *modules có thể xuất/nhập các biến từ file khác vào*
4. *hỗ trợ bất đồng bộ*
5. *mở rộng mảng/đối tượng*
6. *trích xuất/gán biến vào mảng*
7. *iterator/Generator*

**NOTE:**

**code**

[1, 2].forEach((value) => console.log(value));

mảng, vòng lặp each

console.log(name ?? "Người dùng ẩn danh"); *// Người dùng ẩn danh*

trả về giá trị khác null/undefined, ưu tiên vế 1

console.log(length || 10); *// 10*

console.log(length ?? 10); *// 0*

trả về giá trị true (khác rỗng/0/””/null/undefined)